Translate Toolkit Documentation

Release 3.1.1

Translate.org.za

Contents

1	User	's Guide	3
	1.1	Features	3
	1.2	Installation	4
	1.3	Converters	6
	1.4	Tools	60
	1.5	Scripts	99
	1.6	Use Cases	111
	1.7	Translation Related File Formats	128
2	Deve	eloper's Guide	161
	2.1	Translate Styleguide	161
	2.2	Documentation	168
	2.3	Building	171
	2.4	Testing	172
	2.5	Command Line Functional Testing	
	2.6	Contributing	176
	2.7	Translate Toolkit Developers Guide	
	2.8	Making a Translate Toolkit Release	
	2.9	Deprecation of Features	
3	Addi	itional Notes	189
	3.1	Release Notes	189
	3.2	History of the Translate Toolkit	232
	3.3	License	
4	API	Reference	235
	4.1	API	235
Py	thon I	Module Index	733
In	dex		737

Welcome 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 3.8. Install virtualenvwrapper-win to simplify handling of virtualenvs.

- 1. Install latest Python 3.8
- 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[recommended] to install latest stable or pip install –pre translate-toolkit[recommended] to try a pre-release
- 6. 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 3.1.1
```

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/python3.8/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 oo2po 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 l10n/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 l10n/en-US po/xh l10n/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

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

Examples

These examples demonstrate most of the useful invocations of oo2po:

Creating POT files

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

1.3. Converters 11

--multifile=MULTIFILESTYLE how to split po/pot files (single, toplevel or one file)

Extract messages from en-US.sdf and place them in a directory called pot. The -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></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 -1 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:

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

Options:

-P, --pot

--tm=TM

--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 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

1.3. Converters 25

The file to use as translation memory when fuzzy matching

output PO Templates (.pot) rather than PO files (.po)

- -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.

You can create and update PO files for different languages using the pot2po command.

```
po2html -t site -i xh -o 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*.

Notes

The *HTML format description* gives more details on the format of the localisable HTML content and the capabilities of this converter.

Bugs

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

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 <code>json/</code> directory are converted to Gettext POT files and placed in the <code>pot/</code> 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> 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')
Options (po2rc):
```

show program's version number and exit

--version

-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

PD 0 0 PD 0 0 1

--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:

```
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.

 $The \ Symbian \ translation \ header \ field \ \texttt{Author} \ is \ used \ to \ set \ the \ \texttt{Last-Translator} \ PO \ header \ field.$

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.php 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 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.

1.3.28 -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_
    recursiveprocess
    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_
    convertcsv
    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")
```

(continues on next page)

(continued from previous page)

ValueError: An error occurred processing PO file

1.3.29 -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 ""
```

msgctxt (default)

1.3. Converters

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
```

(continues on next page)

55

(continued from previous page)

```
msgctxt "manual_process"
msgid "Manual"
msgstr ""
```

1.3.30 -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.

```
$ oo2po -P --progress=dots en-US.sdf pot

......
$
```

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
```

(continues on next page)

(continued from previous page)

```
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%
```

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
```

1.3.31 -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.

1.3.32 -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.

1.3.33 -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.

1.3.34 -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.

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 -1 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	
Ī	<out></out>	the output file or directory that contains PO or XLIFF files that fail the various tests	

Options:

1.4. Tools 61

--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)

--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 -l
```

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.

1.4. Tools 63

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
 - compendium conflicts, credits, hassuggestion, isfuzzy, isreview, untranslated

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.

1.4. Tools 65

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.

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.

1.4. Tools 67

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 --musttranslatefile.

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. 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, xlff 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

```
tmserver.py --bind=HOSTNAME --port=PORT [--tmdb=TMDBFILE] [--import-translation-

--file=TMFILE [--import-source-lang=SOURCE_LANG] [--import-target-lang=TARGET_LANG]]
```

Where:

TMDB-	is the SQLite database file containing translation memory data, if not specified a new temporary
FILE	database is created
TMFILE	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

```
# apply title-case folding to words before comparing with this stoplist
!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 *.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
type
                             words (source)
                                              words (translation)
                strings
translated: 73465 ( 99%)
                                                  513296
                             538598 ( 99%)
               13 ( 0%)
                               141 ( 0%)
                                                     n/a
fuzzy:
untranslated:
               53 ( 0%)
                               602 ( 0%)
                                                     n/a
Total:
            73531
                             539341
                                                   513296
```

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.:

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]
<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=PROGR	ESS show progress as: dots, none, bar, names, verbose
errorlevel=ERROl	RLEVEL 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
С	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 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, 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

Removes units where sentence number does not correspond

--only-aligned

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)
<pre><current> is the (optional) current English based translation in your intended target language</current></pre>	
<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.

1.5. Scripts 99

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:

BUILD_DIE	The base build directory from where building is done.
MOZCENTRA	ITherefrectory 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_I	The output directory for POT archives.
POUPDATE	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_VERSION	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 110n-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, --110n-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

1.5. Scripts 101

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.

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 b	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.

1.5. Scripts 103

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

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

1.5. Scripts 105

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
    -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:

po is a directory of existing PO files 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

1.5. Scripts 107

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

-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.

1.5. Scripts 109

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

Usage

```
posplit ./file.po
```

Where:

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 -1
```

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_

→files errors
```

Any messages that fail one of the 4 checks will be placed in files in *errors*. We also used the <code>--openoffice</code> 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 <qsi> <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 -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.

<pre>\$ cd af-conflicts \$ ls</pre>					
\$ IS					
change.po	disc.po	functionality.po	letter.po	overwrite.po	
→ restored.po					
changes.po	document.po	gb.po	library.po	page.po	
→ restore.po					
character.po	dots.po	graphic.po	light.po	pager.po	
→ 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					
circle.po	drawing.po	group.po			
etc					

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
                                             lisensieooreenkoms.po paragraaf.po
dataveld.po
                         grys.po

→ sny.po

                         hallo.po
                                             lysinhoud.po
datumgekoop.po
                                                                    pasmaak.po
→ soek.po
                         hiperboliese.po
                                             maateenheid.po
datum.po
                                                                    persentasie.po
→ sorteer.po
                         hoekbeheer.po
                                             maatskappynaam.po
                                                                    posadres.po
deaktiveer.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:

```
\verb|-/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 positive 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

```
\label{lem:msgcat} \begin{array}{ll} {\tt msgcat\ zulu-gnome-glossary-conflicts.po\ >\ zulu-gnome-glossary.po} \end{array}
```

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 -1 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 msqctxt.

```
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 <, & - this doesn't break anything but it would be nicer to see © rather than ©

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
bar
EOT;</pre>
```

• 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';</pre>
```

(continues on next page)

(continued from previous page)

```
// Double slash one-line comment
$messages['file'] = 'File';

/*
    Multi-line
    comment
*/
$messages['help'] = 'Help';
```

• 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.

```
<?php
$lang = array(
    'name' => 'value',
    'datetime' => array(
        'TODAY' => 'Today',
        'YESTERDAY' => 'Yesterday',
        'AGO' => array(
            0 => 'less than a minute ago',
            2 => '%d minutes ago',
            60 => 'l hour ago',
        ),
        'Converted' => 'Converted',
        'LAST' => 'last',
    ),
);
```

• 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.

```
<?php
$variable = array( 'item1' => 'value1', 'item2' => 'value2', 'item3' => 'value3'_

→);
```

· Keyless arrays:

New in version 2.3.0.

• 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>
```

• Keyless arrays assigned to another array:

```
<?php
$messages['days_short'] = array('Sun', 'Mon', 'Tue', 'Wed', 'Thu', 'Fri', 'Sat');</pre>
```

• Laravel plurals are supported in the LaravelPHPFile class:

```
<?php
return [
    'apples' => 'There is one apple|There are many apples',
];
```

Non-Conformance

The following are not yet supported:

• There are currently no known limitations.

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.

The is a tab separated value (TSV) file, where each line represents a translatable unit. A line consists of four elements:

Col-	Description
umn	
source	The source text (in English)
con-	The context of where the source text is used.
text	
re-	An additional remark by the developer, that gives a hint to the translator. Within the context of this toolkit,
marks	this is stored as the note of the unit.
target	The target text

The first line of the file is the header file, with four tab separated values:

- The version (currently: 1)
- The name of the language in lower case (for example: catalan)
- The signature (for example: x-vnd.Haiku-StyledEdit)
- A checksum (32 bit unsigned integer)

The checksum is calculated by an algorithm that hashes the source, context and remark values of all units. The target text is not relevant for the checksum algorithm.

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 prefixed 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)
- Ot 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.

Conformance

- Can identify almost all HTML elements and attributes that are localisable.
- The localisable and localised text in the PO/POT files is fragments of HTML. Therefore, reserved characters must be represented by HTML entities:
 - Content from HTML elements uses the HTML entities & amp; (&), & lt; (<), and & gt; (>).
 - Content from HTML attributes uses the HTML entities " (") or ' (').
- Leading and trailing tags are removed from the localisable text, but only in matching pairs.
- Can cope with embedded PHP, as long as the documents remain valid HTML. If you place PHP code inside
 HTML attributes, you need to make sure that the PHP doesn't contain special characters that interfere with the
 HTML.

References

- · Reserved characters
- Using character entities

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"
]
```

Following JSON dialects are supported

- · Plain JSON files.
- i18next
- Web Extension i18n
- go-i18n
- ARB

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):

· 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 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:

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

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 al 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
-- 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:

(continues on next page)

(continued from previous page)

```
<target>blorp</target>
</trans-unit>
</file>
</xliff>
```

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:
    """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):
    ...
@property
```

(continues on next page)

(continued from previous page)

```
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:

```
@property
def units(self):
# Also good.
@property
def x(self):
 """I'm the 'x' property."""
 return self._x
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 169

(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:

```
#
... 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:
    #: 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 171

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 173

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 conventions
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		\$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 177

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
- tests/cli/data/test_pofilter_manpage/stdout.txt

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 virtualeny. 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.gz
```

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
```

(continues on next page)

(continued from previous page)

```
$ ./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/projects/translate-toolkit/versions/

Use the admin pages to flag a version that should be published.

Note: The branches like stable/2.2.x are automatically enabled on Read the Docs using *Automation Rules*, so there might be nothing to do here.

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 _config.yml and git grep \$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. Tweet about the release.
- 2. Post link to release Tweet to the Translate gitter channel.
- 3. 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 3.1.1

Released on 23 September 2020

This release contains bug fixes.

Changes

General

- Fixed comparing units
- Fixed removal of units from a storage

Contributors

This release was made possible by the following people:

Michal Čihař

And to all our bug finders and testers, a Very BIG Thank You.

Translate Toolkit 3.1.0

Released on 21 September 2020

This release contains improvements and bug fixes.

Changes

Formats and Converters

- YAML
 - Fix generating blank Ruby storage
 - Fix changing type of an unit
 - Preserve structured IDs on round trip
 - Several serialization fixes
- JSON
 - Preserve structured IDs on round trip
 - Several serialization fixes
- Android
 - Fix round trip of strings with newlines
 - Fix escaping double space with HTML
- CSV
 - Allow to control format autodetection
- XWiki
 - Added support for several XWiki formats
- RC
 - Fix parsing adjancent strings
 - Fix handling empty strings
- po2json
 - Never use empty translation in po2json

General

- Added support for removing units from a storage
- Cleaned up storage index usage
- Updated several dependencies
- Dropped version control integration from storage
- Use LRU cache for imports to improve factory performance

Contributors

This release was made possible by the following people:

Michal Čihař, Simon Urli, wojtek555, Stuart Prescott,

And to all our bug finders and testers, a Very BIG Thank You.

Translate Toolkit 3.0.0

Released on 15 June 2020

This release contains improvements and bug fixes.

Changes

Formats and Converters

- PO
 - Bring line wrapping closer to gettext
- XLIFF
 - Support non numeric ids on plurals in poxliff
- JSON
 - Added support for ARB files
 - Added support for go-i18n files
- Properties
 - Added support for GWT personality
 - Fix round trip of empty values without delimiter
- HTML
 - A makeover of the HTML parsing to fix several issues
- PHP
 - Add support for Laravel plurals
 - Improve round trip of some statements
- · Windows RC

- Rewritten parser using pyparsing
- 120n
 - Dropped support for deprecated format

General

- Dropped support for Python 2.7.
- Python 3.5 or newer is now required.
- Minor docs improvements.
- Several cleanups in code.
- Removed deprecated interfaces: multistring no longer accepts encoding search.segment is removed
 pofile.extractpoline is removed simplify_to_common no longer accepts languages parameter get-source/setsource/gettarget/settarget methods removed from storages
- Updated requirements, lxml is no longer optional.
- · Added and updated tests.
- Optional deps can be specified using pip extras.

Contributors

This release was made possible by the following people:

Michal Čihař, papeh, Yann Diorcet, Nick Schonning, Anders Kaplan, Leandro Regueiro, Javier Alfonso, Julen Ruiz Aizpuru

And to all our bug finders and testers, a Very BIG Thank You.

Translate Toolkit 2.5.1

Released on 25 April 2020

This release contains improvements and bug fixes.

Changes

Formats and Converters

- PO
 - Avoid adding extra space on empty comment lines
 - Several performance improvements
- · Android resources
 - Workaround broken plural handling for languages missing other tag
 - Fix setting rich content with comments
 - Fix setting target and removing markup

- YAML
 - Fixed handling of empty keys
 - Improved round trip preserving of comments and style
- TS
 - Avoid duplicating plurals definition
 - Fixed possible crash on adding new translations
- INI
 - Now supported on Python 3 thanks to iniparse support for it
- JSON
 - Allow usage of BOM in JSON files
- MO
 - Fixed context parsing
 - Fixed tests on big endian machines
- · Catkeys
 - The catkeys format now has support for fingerprint calculation

Languages

• Updated plural definitions to match CLDR 36.

General

- Kept support for Python 2.7.
- Fixed py2exe support on Python 2.7.
- Minor docs improvements.
- Minor cleanups in code.
- Updated requirements.
- Added and updated tests.

Contributors

This release was made possible by the following people:

Michal Čihař, Nick Schonning, Tomáš Chvátal, Niels Sascha Reedijk.

And to all our bug finders and testers, a Very BIG Thank You.

Translate Toolkit 2.5.0

Released on 18 January 2020

This release contains improvements and bug fixes.

Changes

Formats and Converters

- PO
 - Avoid stripping empty lines from comments.
 - Raise error on invalid file content.
 - Fixed handling typecomments with non word chars.
 - Fixed serializing long msgidcomments.
- · Properties
 - Avoid creating comment only units (issue 3928).
 - Fixed saving utf-16 Java files.
- Android resources
 - Document declaration is cloned when adding unit.
 - Fixed parsing plurals with comment.
 - Fixed setting plural with markup.
 - Fixed indentation of markup in translation.
 - Fixed XML entities handling.
- YAML
 - Quotes are preserved.
- TS
 - ts2po converts disambiguation notes and comments.
 - po2ts no longer removes consecutive linebreaks in source and translation.
- web2py
 - @markmin string is no longer copied to the translation.

Languages

· Added Sicilian language checks

General

- Kept support for Python 2.7.
- Added support for Python 3.8.
- Minor docs improvements.
- Minor cleanups in code.
- Updated requirements.
- Added and updated tests.

Contributors

This release was made possible by the following people:

Michal Čihař, Leandro Regueiro, Steve Mokris, Queen Vinyl Darkscratch, Matthias, David Paleino.

And to all our bug finders and testers, a Very BIG Thank You.

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 $\$ \theta$ 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

... 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 Virtual 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

- lxml 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 Virtaal 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 Virtaal 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 & 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.
 />
- 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 "e, %, 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 Virtual 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:

- Ot Phrase Book (.qph)
- Ot .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. $\t = \t value \t \t$. 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 po200 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 '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, accesskey_marker='&')
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='&')
```

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 237

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 239

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 or iterable) - 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.

```
class translate.convert.convert.ConvertOptionParser(formats, usetemplates=False, usepots=False, allow miss-ingtemplate=False, description=None)
```

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 241

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.

$\verb|isrecursive| (file option, file purpose = 'input')$

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 or iterable) – 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 243

```
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 245

```
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.
```

with_traceback()

Exception.with_traceback(tb) - set self.__traceback__ to tb and return self.

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.

Convert a source format unit to a target format unit.

See: http://docs.translatehouse.org/projects/translate-toolkit/en/latest/commands/ical2po.html for examples and usage instructions.

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 the converter.
translate.convert.ini2po.run_converter(input_file,
                                                                output file,
                                                                              template_file=None,
                                                                  duplicatestyle='msgctxt',
                                                  pot=False,
```

json2po

Convert JSON files to Gettext PO localization files.

Wrapper around converter.

See: http://docs.translatehouse.org/projects/translate-toolkit/en/latest/commands/json2po.html for examples and usage instructions.

alect='default')

```
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.
```

4.1. API 247

```
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

convert_unit (input_unit, commenttype)
    Converts a JSON 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 JSON files to a PO file
```

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.

```
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.mozlang2po.run_converter(input_file, output_file, template_file=None, pot=False, duplicatestyle='msgctxt', encoding='utf-8')

Wrapper around converter.
```

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.

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,
                                                                              template_file=None,
                                                               output_file,
                                                 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.

```
class translate.convert.po2ical.po2ical(input_file, output_file, template_file=None, in-
                                                   clude_fuzzy=False, output_threshold=None)
     Convert a PO file and a template iCalendar file to a iCalendar file.
     SourceStoreClass
          alias of translate.storage.pypo.pofile
     TargetStoreClass
          alias of translate.storage.ical.icalfile
     TargetUnitClass
          alias of translate.storage.ical.icalunit
     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.po2ical.run_converter(inputfile, outputfile, templatefile=None, include-
                                                   fuzzy=False, outputthreshold=None)
```

po2ini

Convert Gettext PO localization files to .ini files.

Wrapper around converter.

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
     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 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.

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 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 or iterable) - 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.

```
translate.convert.po2rc.is_iterable_but_not_string(o) Check if object is iterable but not a string.
```

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
```

checkoutputsubdir(options, subdir)

define_option(option)

```
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.
class translate.convert.po2tmx.TmxOptionParser(formats,
                                                                          usetemplates=False,
                                                               pots=False, description=None, archive-
                                                               formats=None)
     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 left-
          over arguments (possibly adjusted, possibly completely new – whatever you like). Default implementation
          just returns the passed-in values; subclasses may override as desired.
```

Checks to see if subdir under options.output needs to be created, creates if neccessary.

Defines the given option, replacing an existing one of the same short name if necessary...

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.

$\textbf{finalizetempoutputfile} \ (options, output \textit{file}, full output path)$

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 or iterable) - 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,
                                                     clude_fuzzy=False,
                                                                                output_threshold=None,
                                                     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.

```
 \textbf{class} \  \, \textbf{translate.convert.po2wordfast.WfOptionParser} ( \textit{formats}, \quad \textit{usetemplates=False}, \\ \textit{usepots=False}, \textit{description=None}, \\ \textit{archiveformats=None} )
```

```
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 or iterable) - 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. \textbf{convertpo} (\textit{inputfile}, \textit{outputfile}, \textit{templatefile}, \textit{sourcelan-guage='en'}, \textit{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.

```
class translate.convert.po2yaml.po2yaml(input_file, output_file, template_file=None, in-
                                                   clude fuzzy=False, output threshold=None)
     Convert a PO file and a template YAML file to a YAML file.
     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.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,
                                              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.
exception translate.convert.prop2po.DiscardUnit
     with traceback()
           Exception.with traceback(tb) – set self. traceback to tb and return self.
translate.convert.prop2po.convertmozillaprop (inputfile, outputfile, templatefile, pot=False,
                                                               duplicatestyle='msgctxt')
     Mozilla specific convertor function
translate.convert.prop2po.convertprop(inputfile, outputfile, templatefile, personality='java',
                                                     pot=False,
                                                                    duplicatestyle='msgctxt',
                                                                                                 encod-
                                                     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.
     fold_gwt_plurals (postore)
           Fold the multiple plural units of a gwt file into a gettext plural.
     mergestore (origpropfile, translatedpropfile)
           converts two .properties files to a .po file. . .
```

rc2po

Convert Windows RC files to Gettext PO localization files.

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, dupli-
catestyle='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, dupli-
catestyle='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.

```
\label{lem:convert} translate.convert.tiki2po.run\_converter(\textit{input\_file}, \textit{output\_file}, \textit{template\_file=None}, \textit{includeunused=False}) \\ Wrapper around converter.
```

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
```

```
{\tt 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.convertts(inputfile, outputfile, templates, pot=False, duplicat-
estyle='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,
                                                                              template file=None,
                                                               output file,
                                                 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.\textbf{convertxliff} (\textit{inputfile}, \textit{outputfile}, \textit{templates}, \textit{duplicatestyle='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,
                                                                              template_file=None,
                                                                output_file,
                                                   pot=False, duplicatestyle='msgctxt')
     Wrapper around converter.
class translate.convert.yaml2po.yaml2po(input_file,
                                                                output_file,
                                                                              template_file=None,
                                                   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.
```

filters

run()

Filters that can be used on translations...

Run the converter.

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 (str1, str2)

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 (str1, str2)

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 (str1, str2)

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 (str1, str2)

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 (str1, str2)

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 (str1, str2)

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 (str1, str2)

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(str1, str2)

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 (str1, str2)

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 (str1, str2)

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 (str1, str2)

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(str1, str2)

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 (str1, str2)

Checks whether escaping is consistent between the two strings.

Checks escapes such as $\n \$ to ensure that if they exist in the original string you also have them in the translation.

filepaths (str1, str2)

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 (str1, str2)

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 (str1, str2)

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(str1, str2)

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 (str1, str2)

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 (str1, str2)

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 (str1, str2)

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 (str1, str2)

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 (str1, str2)

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 (str1, str2)

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 (str1, str2)

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 (str1, str2)

Checks that strings that are purely punctuation are not changed.

This extracts strings like + or – as these usually should not be changed.

pythonbraceformat (str1, str2)

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 (str1, str2)

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 (str1, str2)

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 (str1, str2)

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(str1, str2)

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(str1, str2)

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 (str1, str2)

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 (str1, str2)

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(str1, str2)

Checks whether punctuation at the beginning of the strings match.

Operates as endpunc but you will probably see fewer errors.

startwhitespace (str1, str2)

Checks whether whitespace at the beginning of the strings matches.

As in endwhitespace but you will see fewer errors.

tabs (str1, str2)

Checks whether tabs are consistent between the two strings.

Counts the number of \t tab markers and reports an error if they differ.

unchanged (str1, str2)

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 (str1, str2)

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 (str1, str2)

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 (str1, str2)

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 (str1, str2)

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 (str1, str2)

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. or similar translatable attributes in OpenOffice.org help files.

Object representing the configuration of a checker.

update (otherconfig)

Combines the info in otherconfig into this config object.

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 (str1. str2)

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 (str1, str2)

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 (str1, str2)

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 (str1, str2)

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 (str1, str2)

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 (str1, str2)

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 (str1, str2)

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(str1, str2)

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 (str1, str2)

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 (str1, str2)

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 (str1, str2)

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(str1, str2)

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 (str1, str2)

Checks whether escaping is consistent between the two strings.

Checks escapes such as $\n \$ to ensure that if they exist in the original string you also have them in the translation.

filepaths (str1, str2)

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 (str1, str2)

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 (str1, str2)

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(str1, str2)

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 (str1, str2)

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 (str1, str2)

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 (str1, str2)

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 (str1, str2)

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 (str1, str2)

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 (str1, str2)

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 (str1, str2)

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 (str1, str2)

Checks that strings that are purely punctuation are not changed.

This extracts strings like + or – as these usually should not be changed.

pythonbraceformat (str1, str2)

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 (str1, str2)

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 (str1, str2)

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 (str1, str2)

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 (str1, str2)

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(str1, str2)

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 (str1, str2)

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 (str1, str2)

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(str1, str2)

Checks whether punctuation at the beginning of the strings match.

Operates as endpunc but you will probably see fewer errors.

startwhitespace (str1, str2)

Checks whether whitespace at the beginning of the strings matches.

As in endwhitespace but you will see fewer errors.

tabs (str1, str2)

Checks whether tabs are consistent between the two strings.

Counts the number of \t tab markers and reports an error if they differ.

unchanged (str1, str2)

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 (str1, str2)

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 (str1, str2)

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 (str1, str2)

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 (str1, str2)

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 (str1, str2)

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. 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.

with_traceback()

Exception.with traceback(tb) – set self. traceback to tb and return self.

class translate.filters.checks.GnomeChecker(**kwargs)

accelerators (str1, str2)

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 (str1, str2)

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 (str1, str2)

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 (str1, str2)

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 (str1, str2)

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 (str1, str2)

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 (str1, str2)

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(str1, str2)

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 (str1, str2)

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 (str1, str2)

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 (str1, str2)

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 (str1, str2)

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 (str1, str2)

Checks whether escaping is consistent between the two strings.

Checks escapes such as \n to ensure that if they exist in the original string you also have them in the translation.

filepaths (str1, str2)

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 (str1, str2)

Checks that function names are not translated.

Checks that function names e.g. rgb() or getEntity.Name() are not translated.

gconf (str1, str2)

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 (str1, str2)

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(str1, str2)

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 (str1, str2)

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 (str1, str2)

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 (str1, str2)

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 (str1, str2)

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 (str1, str2)

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 (str1, str2)

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 (str1, str2)

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 (str1, str2)

Checks that strings that are purely punctuation are not changed.

This extracts strings like + or - as these usually should not be changed.

pythonbraceformat (str1, str2)

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 (str1, str2)

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 (str1, str2)

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 (str1, str2)

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(str1, str2)

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(str1, str2)

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 (str1, str2)

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 (str1, str2)

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(str1, str2)

Checks whether punctuation at the beginning of the strings match.

Operates as endpunc but you will probably see fewer errors.

startwhitespace (str1, str2)

Checks whether whitespace at the beginning of the strings matches.

As in endwhitespace but you will see fewer errors.

tabs (str1, str2)

Checks whether tabs are consistent between the two strings.

Counts the number of \t tab markers and reports an error if they differ.

unchanged (str1, str2)

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 (str1, str2)

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 (str1, str2)

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 (str1, str2)

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 (str1, str2)

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 (str1, str2)

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. or similar translatable attributes in OpenOffice.org help files.

class translate.filters.checks.IOSChecker(**kwargs)

accelerators (str1, str2)

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 (str1, str2)

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 (str1, str2)

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 (str1, str2)

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 (str1, str2)

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 (str1, str2)

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 (str1, str2)

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(str1, str2)

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 (str1, str2)

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 (str1, str2)

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 (str1, str2)

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 (str1, str2)

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 (str1, str2)

Checks whether escaping is consistent between the two strings.

Checks escapes such as $\n \$ to ensure that if they exist in the original string you also have them in the translation.

filepaths (str1, str2)

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 (str1, str2)

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.

$\verb|getfilters|| (exclude filters = None, limit filters = None)|$

Returns dictionary of available filters, including/excluding those in the given lists.

kdecomments (str1, str2)

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(str1, str2)

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 (str1, str2)

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 (str1, str2)

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 (str1, str2)

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 (str1, str2)

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 (str1, str2)

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 (str1, str2)

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 (str1, str2)

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 (str1, str2)

Checks that strings that are purely punctuation are not changed.

This extracts strings like + or - as these usually should not be changed.

pythonbraceformat (str1, str2)

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 (str1, str2)

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 (str1, str2)

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 (str1, str2)

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(str1, str2)

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(str1, str2)

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 (str1, str2)

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 (str1, str2)

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(str1, str2)

Checks whether punctuation at the beginning of the strings match.

Operates as endpunc but you will probably see fewer errors.

startwhitespace (str1, str2)

Checks whether whitespace at the beginning of the strings matches.

As in endwhitespace but you will see fewer errors.

tabs (str1, str2)

Checks whether tabs are consistent between the two strings.

Counts the number of \t tab markers and reports an error if they differ.

unchanged (str1, str2)

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 (str1, str2)

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 (str1, str2)

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 (str1, str2)

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 (str1, str2)

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 (str1, str2)

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. or similar translatable attributes in OpenOffice.org help files.

class translate.filters.checks.KdeChecker(**kwargs)

accelerators (str1, str2)

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 (str1, str2)

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 (str1, str2)

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 (str1, str2)

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(str1, str2)

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 (str1, str2)

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 (str1, str2)

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(str1, str2)

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 (str1, str2)

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 (str1, str2)

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 (str1, str2)

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(str1, str2)

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 (str1, str2)

Checks whether escaping is consistent between the two strings.

Checks escapes such as $\n \$ to ensure that if they exist in the original string you also have them in the translation.

filepaths (str1, str2)

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 (str1, str2)

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 (str1, str2)

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(str1, str2)

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 (str1, str2)

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 (str1, str2)

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 (str1, str2)

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 (str1, str2)

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 (str1, str2)

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 (str1, str2)

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(str1, str2)

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 (str1, str2)

Checks that strings that are purely punctuation are not changed.

This extracts strings like + or – as these usually should not be changed.

pythonbraceformat (str1, str2)

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 (str1, str2)

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 (str1, str2)

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 (str1, str2)

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 (str1, str2)

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(str1, str2)

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 (str1, str2)

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 (str1, str2)

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(str1, str2)

Checks whether punctuation at the beginning of the strings match.

Operates as endpunc but you will probably see fewer errors.

startwhitespace (str1, str2)

Checks whether whitespace at the beginning of the strings matches.

As in endwhitespace but you will see fewer errors.

tabs (str1, str2)

Checks whether tabs are consistent between the two strings.

Counts the number of \t tab markers and reports an error if they differ.

unchanged (str1, str2)

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 (str1, str2)

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 (str1, str2)

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 (str1, str2)

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 (str1, str2)

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 (str1, str2)

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. or similar translatable attributes in OpenOffice.org help files.

class translate.filters.checks.L20nChecker(**kwargs)

accelerators (str1, str2)

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 (str1, str2)

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 (str1, str2)

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 (str1, str2)

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 (str1, str2)

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 (str1, str2)

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 (str1, str2)

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(str1, str2)

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(str1, str2)

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 (str1, str2)

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 (str1, str2)

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 (str1, str2)

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 (str1, str2)

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 (str1, str2)

Checks whether escaping is consistent between the two strings.

Checks escapes such as $\n \$ to ensure that if they exist in the original string you also have them in the translation.

filepaths (str1, str2)

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 (str1, str2)

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 (str1, str2)

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(str1, str2)

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 (str1, str2)

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 (str1, str2)

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 (str1, str2)

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 (str1, str2)

Checks that numbers are not translated.

Special handling for Mozilla to ignore entries that are dialog sizes.

options (str1, str2)

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(str1, str2)

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 (str1, str2)

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 (str1, str2)

Checks that strings that are purely punctuation are not changed.

This extracts strings like + or - as these usually should not be changed.

pythonbraceformat (str1, str2)

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 (str1, str2)

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 (str1, str2)

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 (str1, str2)

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(str1, str2)

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(str1, str2)

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 (str1, str2)

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 (str1, str2)

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(str1, str2)

Checks whether punctuation at the beginning of the strings match.

Operates as endpunc but you will probably see fewer errors.

startwhitespace (str1, str2)

Checks whether whitespace at the beginning of the strings matches.

As in endwhitespace but you will see fewer errors.

tabs (str1, str2)

Checks whether tabs are consistent between the two strings.

Counts the number of \t tab markers and reports an error if they differ.

unchanged (str1, str2)

Checks whether a translation is basically identical to the original string.

Special handling for Mozilla to ignore entries that are dialog sizes.

untranslated (str1, str2)

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 (str1, str2)

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 (str1, str2)

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 (str1, str2)

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 (str1, str2)

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. or similar translatable attributes in OpenOffice.org help files.

class translate.filters.checks.LibreOfficeChecker(**kwargs)

accelerators (str1, str2)

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 (str1, str2)

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 (str1, str2)

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 (str1, str2)

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(str1, str2)

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 (str1, str2)

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(str1, str2)

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 (str1, str2)

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 (str1, str2)

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 (str1, str2)

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 (str1, str2)

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 (str1, str2)

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 (str1, str2)

Checks whether escaping is consistent between the two strings.

Checks escapes such as \n to ensure that if they exist in the original string you also have them in the translation.

filepaths (str1, str2)

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 (str1, str2)

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 (str1, str2)

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(str1, str2)

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 (str1, str2)

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 (str1, str2)

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 (str1, str2)

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 (str1, str2)

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 (str1, str2)

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 (str1, str2)

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 (str1, str2)

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 (str1, str2)

Checks that strings that are purely punctuation are not changed.

This extracts strings like + or – as these usually should not be changed.

pythonbraceformat (str1, str2)

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 (str1, str2)

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 (str1, str2)

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 (str1, str2)

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(str1, str2)

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(str1, str2)

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 (str1, str2)

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 (str1, str2)

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(str1, str2)

Checks whether punctuation at the beginning of the strings match.

Operates as endpunc but you will probably see fewer errors.

startwhitespace (str1, str2)

Checks whether whitespace at the beginning of the strings matches.

As in endwhitespace but you will see fewer errors.

tabs (str1, str2)

Checks whether tabs are consistent between the two strings.

Counts the number of \t tab markers and reports an error if they differ.

unchanged (str1, str2)

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 (str1, str2)

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 (str1, str2)

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 (str1, str2)

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 (str1, str2)

Check that all XML/HTML open/close tags has close/open pair in the translation.

variables (str1, str2)

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 (str1, str2)

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. or similar translatable attributes in OpenOffice.org help files.

class translate.filters.checks.MinimalChecker(**kwargs)

accelerators (str1, str2)

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 (str1, str2)

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 (str1, str2)

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 (str1, str2)

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 (str1, str2)

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 (str1, str2)

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 (str1, str2)

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(str1, str2)

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 (str1, str2)

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 (str1, str2)

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 (str1, str2)

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 (str1, str2)

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 (str1, str2)

Checks whether escaping is consistent between the two strings.

Checks escapes such as $\n \$ to ensure that if they exist in the original string you also have them in the translation.

filepaths (str1, str2)

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 (str1, str2)

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 (str1, str2)

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(str1, str2)

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 (str1, str2)

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 (str1, str2)

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 (str1, str2)

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 (str1, str2)

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 (str1, str2)

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 (str1, str2)

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 (str1, str2)

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 (str1, str2)

Checks that strings that are purely punctuation are not changed.

This extracts strings like + or - as these usually should not be changed.

pythonbraceformat (str1, str2)

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 (str1, str2)

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 (str1, str2)

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 (str1, str2)

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(str1, str2)

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(str1, str2)

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 (str1, str2)

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 (str1, str2)

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(str1, str2)

Checks whether punctuation at the beginning of the strings match.

Operates as endpunc but you will probably see fewer errors.

startwhitespace (str1, str2)

Checks whether whitespace at the beginning of the strings matches.

As in endwhitespace but you will see fewer errors.

tabs (str1, str2)

Checks whether tabs are consistent between the two strings.

Counts the number of \t tab markers and reports an error if they differ.

unchanged (str1, str2)

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 (str1, str2)

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 (str1, str2)

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 (str1, str2)

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 (str1, str2)

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 (str1, str2)

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. or similar translatable attributes in OpenOffice.org help files.

class translate.filters.checks.MozillaChecker(**kwargs)

accelerators (str1, str2)

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 (str1, str2)

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 (str1, str2)

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 (str1, str2)

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 (str1, str2)

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 (str1, str2)

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 (str1, str2)

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 (str1, str2)

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 (str1, str2)

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 (str1, str2)

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 (str1, str2)

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 (str1, str2)

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(str1, str2)

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 (str1, str2)

Checks whether escaping is consistent between the two strings.

Checks escapes such as $\n \$ to ensure that if they exist in the original string you also have them in the translation.

filepaths (str1, str2)

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 (str1, str2)

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 (str1, str2)

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(str1, str2)

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 (str1, str2)

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 (str1, str2)

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 (str1, str2)

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 (str1, str2)

Checks that numbers are not translated.

Special handling for Mozilla to ignore entries that are dialog sizes.

options (str1, str2)

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(str1, str2)

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 (str1, str2)

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 (str1, str2)

Checks that strings that are purely punctuation are not changed.

This extracts strings like + or – as these usually should not be changed.

pythonbraceformat (str1, str2)

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 (str1, str2)

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 (str1, str2)

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 (str1, str2)

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 (str1, str2)

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(str1, str2)

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 (str1, str2)

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 (str1, str2)

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 (str1, str2)

Checks whether punctuation at the beginning of the strings match.

Operates as endpunc but you will probably see fewer errors.

startwhitespace (str1, str2)

Checks whether whitespace at the beginning of the strings matches.

As in endwhitespace but you will see fewer errors.

tabs (str1, str2)

Checks whether tabs are consistent between the two strings.

Counts the number of \t tab markers and reports an error if they differ.

unchanged (str1, str2)

Checks whether a translation is basically identical to the original string.

Special handling for Mozilla to ignore entries that are dialog sizes.

untranslated (str1, str2)

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 (str1, str2)

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 (str1, str2)

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 (str1, str2)

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 (str1, str2)

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. or similar translatable attributes in OpenOffice.org help files.

class translate.filters.checks.OpenOfficeChecker(**kwargs)

accelerators (str1, str2)

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 (str1, str2)

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 (str1, str2)

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 (str1, str2)

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(str1, str2)

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 (str1, str2)

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(str1, str2)

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(str1, str2)

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 (str1, str2)

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 (str1, str2)

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 (str1, str2)

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(str1, str2)

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 (str1, str2)

Checks whether escaping is consistent between the two strings.

Checks escapes such as $\n \$ to ensure that if they exist in the original string you also have them in the translation.

filepaths (str1, str2)

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 (str1, str2)

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 (str1, str2)

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(str1, str2)

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 (str1, str2)

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 (str1, str2)

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 (str1, str2)

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 (str1, str2)

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 (str1, str2)

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(str1, str2)

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 (str1, str2)

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 (str1, str2)

Checks that strings that are purely punctuation are not changed.

This extracts strings like + or - as these usually should not be changed.

pythonbraceformat (str1, str2)

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 (str1, str2)

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 (str1, str2)

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 (str1, str2)

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(str1, str2)

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(str1, str2)

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 (str1, str2)

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 (str1, str2)

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(str1, str2)

Checks whether punctuation at the beginning of the strings match.

Operates as endpunc but you will probably see fewer errors.

startwhitespace (str1, str2)

Checks whether whitespace at the beginning of the strings matches.

As in endwhitespace but you will see fewer errors.

tabs (str1, str2)

Checks whether tabs are consistent between the two strings.

Counts the number of \t tab markers and reports an error if they differ.

unchanged (str1, str2)

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 (str1, str2)

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 (str1, str2)

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 (str1, str2)

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 (str1, str2)

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 (str1, str2)

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. or similar translatable attributes in OpenOffice.org help files.

class translate.filters.checks.ReducedChecker(**kwargs)

accelerators (str1, str2)

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 (str1, str2)

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 (str1, str2)

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 (str1, str2)

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 (str1, str2)

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 (str1, str2)

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 (str1, str2)

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(str1, str2)

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 (str1, str2)

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 (str1, str2)

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 (str1, str2)

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 (str1, str2)

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 (str1, str2)

Checks whether escaping is consistent between the two strings.

Checks escapes such as $\n \$ to ensure that if they exist in the original string you also have them in the translation.

filepaths (str1, str2)

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 (str1, str2)

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 (str1, str2)

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(str1, str2)

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 (str1, str2)

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 (str1, str2)

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 (str1, str2)

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 (str1, str2)

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 (str1, str2)

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(str1, str2)

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 (str1, str2)

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 (str1, str2)

Checks that strings that are purely punctuation are not changed.

This extracts strings like + or - as these usually should not be changed.

pythonbraceformat (str1, str2)

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 (str1, str2)

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 (str1, str2)

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 (str1, str2)

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 (str1, str2)

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(str1, str2)

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 (str1, str2)

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 (str1, str2)

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(str1, str2)

Checks whether punctuation at the beginning of the strings match.

Operates as endpunc but you will probably see fewer errors.

startwhitespace (str1, str2)

Checks whether whitespace at the beginning of the strings matches.

As in endwhitespace but you will see fewer errors.

tabs (str1, str2)

Checks whether tabs are consistent between the two strings.

Counts the number of \t tab markers and reports an error if they differ.

unchanged (str1, str2)

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 (str1, str2)

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 (str1, str2)

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 (str1, str2)

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 (str1, str2)

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 (str1, str2)

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. 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)

with traceback()

Exception.with_traceback(tb) - set self.__traceback__ to tb and return self.

The basic test suite for source -> target translations.

accelerators (str1, str2)

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 (str1, str2)

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 (str1, str2)

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 (str1, str2)

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 (str1, str2)

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 (str1, str2)

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 (str1, str2)

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 (str1, str2)

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 (str1, str2)

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 (str1, str2)

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 (str1, str2)

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 (str1, str2)

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 (str1, str2)

Checks whether escaping is consistent between the two strings.

Checks escapes such as $\n \$ to ensure that if they exist in the original string you also have them in the translation.

filepaths (str1, str2)

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 (str1, str2)

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.

$\verb|getfilters|| (exclude filters = None, limit filters = None)|$

Returns dictionary of available filters, including/excluding those in the given lists.

kdecomments (str1, str2)

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(str1, str2)

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 (str1, str2)

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 (str1, str2)

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 (str1, str2)

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 (str1, str2)

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 (str1, str2)

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 (str1, str2)

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 (str1, str2)

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 (str1, str2)

Checks that strings that are purely punctuation are not changed.

This extracts strings like + or – as these usually should not be changed.

pythonbraceformat (str1, str2)

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 (str1, str2)

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 (str1, str2)

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 (str1, str2)

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(str1, str2)

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(str1, str2)

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 (str1, str2)

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 (str1, str2)

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(str1, str2)

Checks whether punctuation at the beginning of the strings match.

Operates as endpunc but you will probably see fewer errors.

startwhitespace (str1, str2)

Checks whether whitespace at the beginning of the strings matches.

As in endwhitespace but you will see fewer errors.

tabs (str1, str2)

Checks whether tabs are consistent between the two strings.

Counts the number of \t tab markers and reports an error if they differ.

unchanged (str1, str2)

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 (str1, str2)

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 (str1, str2)

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 (str1, str2)

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 (str1, str2)

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 (str1, str2)

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. or similar translatable attributes in OpenOffice.org help files.

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.

```
\verb|getfilters|| \textit{excludefilters} = None, \textit{limitfilters} = None)
```

Returns dictionary of available filters, including/excluding those in the given lists.

hassuggestion (unit)

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(unit)

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(unit)

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 (unit)

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 (str1, str2)

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 (str1, str2)

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 (str1, str2)

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 (str1, str2)

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 (str1, str2)

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 (str1, str2)

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(str1, str2)

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(str1, str2)

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 (str1, str2)

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 (str1, str2)

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 (str1, str2)

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(str1, str2)

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 (str1, str2)

Checks whether escaping is consistent between the two strings.

Checks escapes such as $\n \$ to ensure that if they exist in the original string you also have them in the translation.

filepaths (str1, str2)

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 (str1, str2)

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 (str1, str2)

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(str1, str2)

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 (str1, str2)

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 (str1, str2)

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 (str1, str2)

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 (str1, str2)

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 (str1, str2)

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 (str1, str2)

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(str1, str2)

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 (str1, str2)

Checks that strings that are purely punctuation are not changed.

This extracts strings like + or - as these usually should not be changed.

pythonbraceformat (str1, str2)

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 (str1, str2)

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 (*str1*, *str2*)

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 (str1, str2)

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 (str1, str2)

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(str1, str2)

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 (str1, str2)

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 (str1, str2)

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(str1, str2)

Checks whether punctuation at the beginning of the strings match.

Operates as endpunc but you will probably see fewer errors.

startwhitespace (str1, str2)

Checks whether whitespace at the beginning of the strings matches.

As in endwhitespace but you will see fewer errors.

tabs (str1, str2)

Checks whether tabs are consistent between the two strings.

Counts the number of \t tab markers and reports an error if they differ.

unchanged (str1, str2)

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 (str1, str2)

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 (str1, str2)

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 (str1, str2)

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 (str1, str2)

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 (str1, str2)

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. 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.getvariables(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(str1)
    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(str1, 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,...)
```

build_checkerconfig(options)

Prepare the checker config from the given options. This is mainly factored out for the sake of unit tests.

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 or iterable) – 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.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)
```

4.1. API 355

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)

 $\verb|translate.filters.prefilters.filtervariables| (startmarker, end marker, var filter)|$

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 = {'': 'Jo', '': 'A', '': 'B', '': 'W', '': 'G', '': 'D', '': 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 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.

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
- Quotes
 - 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 = ''

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 = ''

The ISO 639 language code, possibly with a country specifier or other modifier.

Examples:

```
km
pt_BR
sr_YU@Latn
```

commonpunc = '.,;:!?-@#\$%^*_()[]{}/\\'`"<>'

These punctuation marks are common in English and most languages that use latin script.

ethiopicpunc = ''

These punctuation marks are used by several Ethiopic languages.

fullname = ''

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 = ''

These punctuation marks are used by several Indic languages.

invertedpunc = '¿;'

Inverted punctuation sometimes used at the beginning of sentences in Spanish, Asturian, Galician, and Catalan.

classmethod length_difference(length)

Returns an estimate to a likely change in length relative to an English string of length length.

listseperator = ', '

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 = '...±°123.©®×£¥€'

The middle dot (\cdot) is used by Greek and Georgian.

mozilla pluralequation = '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 = '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 = '.,;:!?-@#$%^*_()[]{}/\\'`"<>`'"",< >«»¿¡÷...±°123 ·©®×£¥€'
```

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.

```
quotes = ' ''"", < > < > '
```

These are different quotation marks used by various languages.

rtlpunc = '÷'

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.

sentenceend = '.!?...'

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 = ''
```

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 = ['zero', 'one', 'two', 'few', 'many', 'other']
List of plural tags generated from CLDR 32.0.1 using https://github.com/WeblateOrg/language-data
```

```
translate.lang.data.expansion_factors = {'af': 0.1, 'ar': -0.09, 'es': 0.21, 'fr': 0.26 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.

```
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.

```
\verb|translate.lang.data.languagematch| (languagecode, other languagecode)|
```

matches a languagecode to another, ignoring regions in the second

translate.lang.data.languages = {'ach': ('Acholi', 2, 'n > 1'), 'af': ('Afrikaans', 2, '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='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 = {'Beng': ['bn', 'mni'], 'Deva': ['anp', 'bho', 'brx', 'doi 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)
```

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.

${\tt classmethod\ length_difference}\ (\mathit{length})$

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.

${\tt classmethod\ punctranslate}\ ({\it text})$

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
```

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.

${\tt classmethod\ length_difference}\ (\mathit{length})$

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)

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.

${\tt classmethod\ punctranslate}\ (\textit{text})$

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.

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.

${\tt classmethod\ length_difference}\ (\mathit{length})$

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.

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 = ''

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.

classmethod words(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.

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.

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.

```
translate.lang.poedit.dialects = {'Chinese': {'CHINA': 'zh_CN', 'None': 'zh_CN', 'TAIWAN' Language dialects based on ISO 3166 country names, 'None' is the default fallback
```

```
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:

- 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 = {'aa': 'Afar', 'ab': 'Abkhazian', 'ae': 'Avestan', 'a 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 = {'(Afan) Oromo': 'om', 'Abkhazian': 'ab', 'Afar': 'a
    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.

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.

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.

${\tt classmethod}$ 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.

${\tt classmethod\ length_difference}\ (length)$

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 = { 'af': ('i18n@af.org.za', 'Petri Jooste')
     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)
```

th

This module represents the Thai language.

Returns a list of words in text.

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.

${\tt classmethod\ length_difference}\ (\mathit{length})$

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.

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 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.

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()
           Create a new dictionary with keys from iterable and values set to value.
      get (key, default=None)
           Return the value for key if key is in the dictionary, else default.
      items () \rightarrow a set-like object providing a view on D's items
      keys () \rightarrow a set-like object providing a view on 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()
           Insert key with a value of default if key is not in the dictionary.
           Return the value for key if key is in the dictionary, else default.
      update (E) \rightarrow None.
           Update D from E: for k in E.keys(): D[k] = E[k]
      values () \rightarrow an object providing a view on D's values
file discovery
translate.misc.file_discovery.qet_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().
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()
           Return a capitalized version of the string.
           More specifically, make the first character have upper case and the rest lower case.
      casefold()
           Return a version of the string suitable for caseless comparisons.
      center()
           Return a centered 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 string S[start:end]. Optional arguments start and end are interpreted as in slice notation.

encode()

Encode the string using the codec registered for encoding.

encoding The encoding in which to encode the string.

errors The error handling scheme to use for encoding errors. The 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()

Return a copy 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) → str

Return a formatted version of S, using substitutions from args and kwargs. The substitutions are identified by braces ('{' and '}').

$format_map(mapping) \rightarrow str$

Return a formatted version of S, using substitutions from mapping. The substitutions are identified by braces ('{ and '}').

$index(sub[, start[, end]]) \rightarrow int$

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.

Raises ValueError when the substring is not found.

isalnum()

Return True if the string is an alpha-numeric string, False otherwise.

A string is alpha-numeric if all characters in the string are alpha-numeric and there is at least one character in the string.

isalpha()

Return True if the string is an alphabetic string, False otherwise.

A string is alphabetic if all characters in the string are alphabetic and there is at least one character in the string.

isascii()

Return True if all characters in the string are ASCII, False otherwise.

ASCII characters have code points in the range U+0000-U+007F. Empty string is ASCII too.

isdecimal()

Return True if the string is a decimal string, False otherwise.

A string is a decimal string if all characters in the string are decimal and there is at least one character in the string.

isdigit()

Return True if the string is a digit string, False otherwise.

A string is a digit string if all characters in the string are digits and there is at least one character in the string.

isidentifier()

Return True if the string is a valid Python identifier, False otherwise.

Use keyword.iskeyword() to test for reserved identifiers such as "def" and "class".

islower()

Return True if the string is a lowercase string, False otherwise.

A string is lowercase if all cased characters in the string are lowercase and there is at least one cased character in the string.

isnumeric()

Return True if the string is a numeric string, False otherwise.

A string is numeric if all characters in the string are numeric and there is at least one character in the string.

isprintable()

Return True if the string is printable, False otherwise.

A string is printable if all of its characters are considered printable in repr() or if it is empty.

isspace()

Return True if the string is a whitespace string, False otherwise.

A string is whitespace if all characters in the string are whitespace and there is at least one character in the string.

istitle()

Return True if the string is a title-cased string, False otherwise.

In a title-cased string, upper- and title-case characters may only follow uncased characters and lowercase characters only cased ones.

isupper()

Return True if the string is an uppercase string, False otherwise.

A string is uppercase if all cased characters in the string are uppercase and there is at least one cased character in the string.

join()

Concatenate any number of strings.

The string whose method is called is inserted in between each given string. The result is returned as a new string.

Example: '.'.join(['ab', 'pq', 'rs']) -> 'ab.pq.rs'

ljust()

Return a left-justified string of length width.

Padding is done using the specified fill character (default is a space).

lower()

Return a copy of the string converted to lowercase.

lstrip()

Return a copy of the string with leading whitespace removed.

If chars is given and not None, remove characters in chars instead.

static maketrans()

Return a translation table usable for str.translate().

If there is only one argument, it must be a dictionary mapping Unicode ordinals (integers) or characters to Unicode ordinals, strings or None. Character keys will be then converted to ordinals. If there are two arguments, they must be strings of equal length, and in the resulting dictionary, each character in x will be mapped to the character at the same position in y. If there is a third argument, it must be a string, whose characters will be mapped to None in the result.

partition()

Partition the string into three parts using the given separator.

This will search for the separator in the string. If the separator is found, returns a 3-tuple containing the part before the separator, the separator itself, and the part after it.

If the separator is not found, returns a 3-tuple containing the original string and two empty strings.

replace (old, new, count=None)

Return a copy with all occurrences of substring old replaced by new.

count Maximum number of occurrences to replace. -1 (the default value) means replace all occurrences.

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.

$rindex(sub[, start[, end]]) \rightarrow int$

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.

Raises ValueError when the substring is not found.

rjust()

Return a right-justified string of length width.

Padding is done using the specified fill character (default is a space).

rpartition()

Partition the string into three parts using the given separator.

This will search for the separator in the string, starting at the end. If the separator is found, returns a 3-tuple containing the part before the separator, the separator itself, and the part after it.

If the separator is not found, returns a 3-tuple containing two empty strings and the original string.

rsplit()

Return a list of the words in the string, using sep as the delimiter string.

sep The delimiter according which to split the string. None (the default value) means split according to any whitespace, and discard empty strings from the result.

maxsplit Maximum number of splits to do. -1 (the default value) means no limit.

Splits are done starting at the end of the string and working to the front.

rstrip()

Return a copy of the string with trailing whitespace removed.

If chars is given and not None, remove characters in chars instead.

split()

Return a list of the words in the string, using sep as the delimiter string.

sep The delimiter according which to split the string. None (the default value) means split according to any whitespace, and discard empty strings from the result.

maxsplit Maximum number of splits to do. -1 (the default value) means no limit.

splitlines()

Return a list of the lines in the string, 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()

Return a copy of the string with leading and trailing whitespace remove.

If chars is given and not None, remove characters in chars instead.

swapcase()

Convert uppercase characters to lowercase and lowercase characters to uppercase.

title()

Return a version of the string where each word is titlecased.

More specifically, words start with uppercased characters and all remaining cased characters have lower case.

translate()

Replace each character in the string using the given translation table.

table Translation table, which must be a mapping of Unicode ordinals to Unicode ordinals, strings, or None.

The table must implement lookup/indexing via __getitem__, for instance a dictionary or list. If this operation raises LookupError, the character is left untouched. Characters mapped to None are deleted.

upper()

Return a copy of the string converted to uppercase.

zfill(

Pad a numeric string with zeros on the left, to fill a field of the given width.

The string is never truncated.

optrecurse

format_option_strings(option)

Return a comma-separated list of option strings & metavariables.

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.

```
class translate.misc.optrecurse.ManPageOption(*opts, **attrs)
     take_action (action, dest, opt, value, values, parser)
           take_action that can handle manpage as well as standard actions
class translate.misc.optrecurse.RecursiveOptionParser(formats, usetemplates=False,
                                                                            allowmissingtemplate=False,
                                                                            description=None)
     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 left-
           over 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...
```

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 or iterable) - 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.Document

documentElement

Top-level element of this document.

firstChild

First child node, or None.

lastChild

Last child node, or None.

localName

Namespace-local name of this node.

attributes

NamedNodeMap of attributes on the element.

firstChild

First child node, or None.

lastChild

Last child node, or None.

localName

Namespace-local name of this element.

class translate.misc.ourdom.ExpatBuilderNS(options=None)

createParser()

Create a new namespace-handling parser.

getParser()

Return the parser object, creating a new one if needed.

install(parser)

Insert the namespace-handlers onto the parser.

parseFile (file)

Parse a document from a file object, returning the document node.

parseString(string)

Parse a document from a string, returning the document node.

reset (

Free all data structures used during DOM construction.

$\verb|start_namespace_decl_handler| (\textit{prefix}, \textit{uri})$

Push this namespace declaration on our storage.

```
translate.misc.ourdom.getElementsByTagName_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.
     show (verbosemessage)
          displays the progress bar
class translate.misc.progressbar.MessageProgressBar(*args, **kwargs)
     A ProgressBar that just writes out the messages without any progress display
     show (verbosemessage)
          displays the progress bar
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
class translate.misc.progressbar.VerboseProgressBar(*args, **kwargs)
     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 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, allowreentry=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, allowreentry=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 substr occurs in searchin locations are not allowed to overlap

```
translate.misc.quote.htmlentitydecode(source)
```

Decode source using HTML entities e.g. © -> ©.

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.

wsqi

```
Wrapper to launch the bundled CherryPy server.
```

```
translate.misc.wsgi.launch_server (host, port, app, **kwargs)
Use cheroot WSGI server, a multithreaded scallable server.
```

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, leaves=None)
```

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:spaceAll 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.

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.

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

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.

```
class translate.storage.base.DictStore (unitclass=None, encoding=None)
```

UnitClass

```
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
removeunit (unit)
     Remove 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.

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.base.DictUnit (source=None)

DefaultDict

alias of collections. Ordered Dict

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.

$\verb"addlocation" (location")$

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()

Returns dictionary for serialization.

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.

${\tt markreviewneeded}\ (needs review=True,\ explanation=None)$

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:

```
>>> target = multistring(['foo', 'bar', 'baz'])
>>> TranslationUnit.multistring_to_rich(target)
[<StringElem([<StringElem(['foo'])>])>,
```

(continues on next page)

(continued from previous page)

```
<StringElem([<StringElem(['bar'])>])>,
           <StringElem([<StringElem(['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()
     \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('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.
exception translate.storage.base.ParseError(inner_exc)
     with_traceback()
          Exception.with_traceback(tb) - set self.__traceback__ to tb and return self.
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

removeunit (unit)

Remove 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.

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.

suggestions_in_format = False

Indicates if format can store suggestions and alternative translation for a unit

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.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.

$\verb|isobsolete|()|$

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 (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(['foo', 'bar', 'baz'])
>>> TranslationUnit.multistring_to_rich(target)
[<StringElem([<StringElem(['foo'])>])>,
    <StringElem([<StringElem(['bar'])>])>,
    <StringElem([<StringElem(['baz'])>])>]
```

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('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',
```

Parameters delete_orig - If True, as set by convert_forward(), afile is deleted after appending, if possible.

Note: For this implementation, the appended file will be deleted from disk if delete_orig is True.

cleanup()

'tgt').

Clean up our mess: remove temporary files.

get_file (fname)

Retrieve a project file (source, translation or target file) from the project archive.

${\tt get_filename_type}\ (\textit{fname})$

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.
              Returns the results from BundleProjStore.append file().
exception translate.storage.bundleprojstore.InvalidBundleError
     with traceback()
          Exception.with_traceback(tb) - set self.__traceback__ to tb and return self.
catkeys
Manage the Haiku catkeys translation format
The Haiku catkeys format is the translation format used for localisation of the Haiku operating system.
It is a bilingual base class derived format with CatkeysFile and CatkeysUnit providing file and unit level
access. The file format is described here: http://www.haiku-os.org/blog/pulkomandy/2009-09-24_haiku_locale_kit_
translator handbook
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.
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
```

4.1. API 415

This method should always be used rather than trying to modify the list manually.

addunit (unit)

Append the given unit to the object's list of units.

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

removeunit (unit)

Remove 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.

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

setchecksum(checksum)

Set the checksum for the file

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.

${\tt markreviewneeded}\ (needs review=True,\ explanation=None)$

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:

```
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('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 = ['source', 'context', 'comment', 'target']
Field names for a catkeys TU
```

```
translate.storage.catkeys.FIELDNAMES_HEADER = ['version', 'language', 'mimetype', 'checksu
Field names for the catkeys header
```

```
translate.storage.catkeys.FIELDNAMES_HEADER_DEFAULTS = {'checksum': '', 'language': ''
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.DefaultDialect
```

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.

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 (csvsrc, sample_length=1024)
           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
     removeunit (unit)
           Remove 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.
     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='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.

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:

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('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

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.

unit_iter()
        Iterator over all the units in all the files in this directory.
```

dtd

Classes that hold units of .dtd files (dtdunit) or entire files (dtdfile).

These are specific .dtd files for localisation used by mozilla.

Specifications The following information is provided by Mozilla:

Specification

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 < (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.

 $Return\ type\ {\tt TranslationUnit}$

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 (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
removeunit (unit)
     Remove 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.
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)

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()

returns whether this dtdunit doesn't actually have an entity definition

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:

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('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 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 or TranslationStore) - 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.
     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.
         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. '
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.

```
remove unit from index(unit)
          Remove a unit from source and locaton indexes
     removeunit (unit)
          Remove 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.
     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)
     EMPTY HTML ELEMENTS = ['area', 'base', 'br', 'col', 'embed', 'hr', 'imq', 'input', 'li
          An empty element is an element that cannot have any child nodes (i.e., nested elements or text nodes). In
          HTML, using a closing tag on an empty element is usually invalid. Reference https://developer.mozilla.
          org/en-US/docs/Glossary/Empty_element
     TRANSLATABLE_ATTRIBUTES = ['abbr', 'alt', 'lang', 'summary', 'title', 'value']
          Text from these HTML attributes will be extracted as translation units. Note: the content attribute of meta
          tags is a special case.
     TRANSLATABLE_ELEMENTS = ['address', 'article', 'aside', 'blockquote', 'caption', 'dd',
          These HTML elements (tags) will be extracted as translation units, unless they lack translatable text con-
          tent. In case one translatable element is embedded in another, the outer translation unit will be split into
          the parts before and after the inner translation unit.
```

4.1. API 435

Document metadata from meta elements with these names will be extracted as translation units. Reference

TRANSLATABLE_METADATA = ['description', 'keywords']

https://developer.mozilla.org/en-US/docs/Web/HTML/Element/meta/name

```
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.
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.
         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. '

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.

${\tt remove_unit_from_index}~(\textit{unit})$

Remove a unit from source and locaton indexes

removeunit (unit)

Remove 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.

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.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)

${\tt classmethod\ buildfromunit}\ (\mathit{unit})$

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:

```
>>> target = multistring(['foo', 'bar', 'baz'])
>>> TranslationUnit.multistring_to_rich(target)
[<StringElem([<StringElem(['foo'])>])>,
    <StringElem([<StringElem(['bar'])>])>,
    <StringElem([<StringElem(['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('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.
```

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:

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)

4.1. API 441

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
removeunit (unit)
     Remove 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.
```

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 <code>getlocations()</code>).

${\tt gettargetlen}\,(\,)$

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:

```
>>> target = multistring(['foo', 'bar', 'baz'])
>>> TranslationUnit.multistring_to_rich(target)
[<StringElem([<StringElem(['foo'])>])>,
    <StringElem([<StringElem(['bar'])>])>,
    <StringElem([<StringElem(['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('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 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.
```

ini

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

removeunit (unit)

Remove 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.

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.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 <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.

${\tt hasplural} \; (\,)$

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('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:

```
"firstName": "John",
    "lastName": "Smith",
    "age": 25,
    "address": {
        "streetAddress": "21 2nd Street",
        "city": "New York",
        "state": "NY",
```

(continues on next page)

(continued from previous page)

TODO:

- Handle \u and other escapes in Unicode
- Manage data type storage and conversion. True -> "True" -> True

```
class translate.storage.jsonl10n.ARBJsonFile(inputfile=None, filter=None, **kwargs)
    ARB JSON file
```

See following URLs for doc:

https://github.com/google/app-resource-bundle/wiki/ApplicationResourceBundleSpecification https://flutter.dev/docs/development/accessibility-and-localization/internationalization#appendix-using-the-dart-intl-tools

UnitClass

alias of ARBJsonUnit

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

removeunit (unit)

Remove 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.

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.

DefaultDict

alias of collections. Ordered Dict

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()

Returns dictionary for serialization.

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 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('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.

go-i18n JSON file

See following URLs for doc:

https://github.com/nicksnyder/go-i18n https://godoc.org/github.com/nicksnyder/go-i18n/v2

UnitClass

alias of GoI18NJsonUnit

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
removeunit (unit)
     Remove 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.
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.

DefaultDict

alias of collections. OrderedDict

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()

Returns dictionary for serialization.

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(['foo', 'bar', 'baz'])
>>> TranslationUnit.multistring_to_rich(target)
[<StringElem([<StringElem(['foo'])>])>,
    <StringElem([<StringElem(['bar'])>])>,
    <StringElem([<StringElem(['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('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.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.

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

removeunit (unit)

Remove 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.

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.

A i18next v3 format, JSON with plurals.

See https://www.i18next.com/

DefaultDict

alias of collections.OrderedDict

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()

Returns dictionary for serialization.

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('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. 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 removeunit (unit) Remove 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. 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.

Return the translated string for a given source string.

translate(source)

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 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()

4.1. API 469

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

removeunit (unit)

Remove 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.

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.

DefaultDict

alias of collections. OrderedDict

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()

Returns dictionary for serialization.

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(['foo', 'bar', 'baz'])
>>> TranslationUnit.multistring_to_rich(target)
[<StringElem([<StringElem(['foo'])>])>,
    <StringElem([<StringElem(['bar'])>])>,
    <StringElem([<StringElem(['baz'])>])>]
```

removenotes (origin=None)

Remove all the translator's notes.

```
rich_source
```

See also:

```
rich_to_multistring(), multistring_to_rich()
```

${\tt rich_target}$

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('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.

DefaultDict

alias of collections. Ordered Dict

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()

Returns dictionary for serialization.

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.

$\verb|istranslatable| ()$

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(['foo', 'bar', 'baz'])
>>> TranslationUnit.multistring_to_rich(target)
[<StringElem([<StringElem(['foo'])>])>,
    <StringElem([<StringElem(['bar'])>])>,
    <StringElem([<StringElem(['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('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

**kwargs)

477

unit iter() Iterator that only returns this unit. class translate.storage.jsonl10n.WebExtensionJsonFile(inputfile=None, filter=None, 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. 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.

Return a list of all units in this store.

getunits()

isempty()

4.1. API

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
     removeunit (unit)
           Remove 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.
     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.
                                                                                            item=None,
class translate.storage.jsonl10n.WebExtensionJsonUnit(source=None,
                                                                           notes=None,
                                                                                             placehold-
                                                                           ers=None, **kwargs)
     DefaultDict
```

alias of collections. Ordered Dict

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()

Returns dictionary for serialization.

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('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.
     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

removeunit(unit)

Remove 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.

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)

${\tt classmethod\ buildfromunit}\ (\mathit{unit})$

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 overridden.

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'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('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, 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.

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

removeunit (unit)

Remove 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.

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 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.

$\verb"adderror" (\textit{errorname}, \textit{errortext})$

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('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

removeunit (unit)

Remove 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.

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.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:

```
>>> target = multistring(['foo', 'bar', 'baz'])
>>> TranslationUnit.multistring_to_rich(target)
[<StringElem([<StringElem(['foo'])>])>,
    <StringElem([<StringElem(['bar'])>])>,
    <StringElem([<StringElem(['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('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 = ['source', 'target', 'comment']
     Field names for an OmegaT glossary unit
class translate.storage.omegat.OmegaTDialect
     Describe the properties of an OmegaT generated TAB-delimited glossary file.
class translate.storage.omegat.OmegaTFile(inputfile=None, **kwargs)
     An OmegaT glossary file
     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

removeunit (unit)

Remove 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.

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)

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

removeunit (unit)

Remove 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.

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 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 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 <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.

 $\textbf{merge} \ (other unit, overwrite = False, comments = True, authoritative = False)$

Do basic format agnostic merging.

multistring_to_rich(mulstring)

Convert a multistring to a list of "rich" string trees:

```
>>> target = multistring(['foo', 'bar', 'baz'])
>>> TranslationUnit.multistring_to_rich(target)
[<StringElem([<StringElem(['foo'])>])>,
```

(continues on next page)

(continued from previous page)

```
<StringElem([<StringElem(['bar'])>])>,
<StringElem([<StringElem(['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('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.

00

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.int2byte()
     S.pack(v1, v2, ...) \rightarrow bytes
     Return a bytes object containing values v1, v2, ... packed according to the format string S.format. See
     help(struct) for more on format strings.
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
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()
           Create a new dictionary with keys from iterable and values set to value.
      get()
           Return the value for key if key is in the dictionary, else default.
      items () \rightarrow a set-like object providing a view on D's items
      keys () \rightarrow a set-like object providing a view on 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()
           Insert key with a value of default if key is not in the dictionary.
```

Return the value for key if key is in the dictionary, else default.

update ([E], **F) \rightarrow None. Update D from dict/iterable E and F.

If E is present and has a .keys() method, then does: for k in E: D[k] = E[k] If E is present and lacks a .keys() method, then 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 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.

class translate.storage.placeables.base.Bpt(sub=None, id=None, rid=None, xid=None, xid=None, rid=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.

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='utf-8')

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.Ept (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 \in starts, or -1 if \in was not found.

encode (encoding='utf-8')

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='utf-8')

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='utf-8')

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='utf-8')

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='utf-8')

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)

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='utf-8')

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='utf-8')

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='utf-8')

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='utf-8')
```

More unicode class emulation.

find(x)

Find sub-string \boldsymbol{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.

```
class translate.storage.placeables.general.XMLEntityPlaceable (sub=None, id=None, rid=None, xid=None, xi
```

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='utf-8')

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='utf-8')

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='utf-8')

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.InvisiblePlaceable (sub=None, id=None, id=None, rid=None, xid=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.

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='utf-8')

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
```

${\tt apply_to_strings}\,(f)$

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='utf-8')

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='utf-8')

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='utf-8')

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

with_traceback()

Exception.with_traceback(tb) - set self.__traceback__ to tb and return self.

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.

elem offset (elem)

Find the offset of elem in the current tree.

This cannot be reliably used if <code>self.renderer</code> is used and even less so if the rendering function renders the string differently upon different calls. In Virtual the <code>StringElemGUI.index()</code> 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='utf-8')

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='utf-8')

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='utf-8')

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='utf-8')

More unicode class emulation.

find(x)

Find sub-string \boldsymbol{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='utf-8')

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.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='utf-8')

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='utf-8')

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='utf-8')

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='utf-8')

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
```

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='utf-8')

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.

 $\textbf{class} \text{ translate.storage.placeables.xliff.Ph} (\textit{sub=None}, \textit{id=None}, \textit{rid=None}, \textit{xid=None}, \textit{x$

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.

${\tt elem_offset}\;(elem)$

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='utf-8')

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='utf-8')

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:

```
<?php
$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.

```
class translate.storage.php.LaravelPHPFile(inputfile=None, **kwargs)
```

```
UnitClass
```

alias of LaravelPHPUnit

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 (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 removeunit (unit) Remove 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. 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)

4.1. API 559

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.LaravelPHPUnit (source=")

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 <code>getlocations()</code>).

getoutput (indent=", 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('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.phpdecode(text, quotechar=""")
     Convert PHP escaped string to a Python string.
translate.storage.php.phpencode (text, quotechar="'")
     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
     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 (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

removeunit (unit)

Remove 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.

require_index()

make sure source index exists

save()

Save to the file that data was originally read from, if available.

${\tt savefile}\,(\mathit{storefile})$

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=")

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.

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()).

getoutput (indent=", 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.

$\verb|istranslatable| ()$

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('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, noheader=False, **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
removeunit (unit)
     Remove 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.
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.
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.

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(['foo', 'bar', 'baz'])
>>> TranslationUnit.multistring_to_rich(target)
[<StringElem([<StringElem(['foo'])>])>,
    <StringElem([<StringElem(['bar'])>])>,
    <StringElem([<StringElem(['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('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.

```
\verb|translate.storage.poparser.read_prevmsgid_lines| (\textit{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.

removeunit (unit)

Remove 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.

```
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.

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 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.

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()
     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('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
     with_traceback()
          Exception.with traceback(tb) – set self. traceback to tb and return self.
exception translate.storage.projstore.FileNotInProjectError
     with traceback()
          Exception.with_traceback(tb) - set self.__traceback__ to tb and return self.
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
- · XWiki .properties

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.

XWiki XWiki translations files are standard Java .properties but with specific escaping support for simple quotes, and support of missing translations. This XWiki document provides the information used to implement the dialect.

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
b = a string with escape sequences <math>\t \n \ \" \" \" \' \ (space) \u0123
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)
# .strings specific
"key" = "value";
```

$\textbf{class} \ \texttt{translate.storage.properties.} \textbf{Dialect}$

Settings for the various behaviours in key=value files.

```
classmethod encode (string, encoding=None)
```

Encode the string

${\tt classmethod\ find_delimiter}\ ({\it line})$

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.DialectGwt

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.DialectJavaUtf16

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)

classmethod key_strip(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

```
class translate.storage.properties.DialectXWiki
```

XWiki dialect is mainly a Java properties behaviour but with special handling of simple quotes: they are escaped by doubling them when an argument on the form "{X}" is provided, X being a number.

classmethod encode (string, encoding=None)

Encode the string

${\tt classmethod\ find_delimiter}\ (\mathit{line})$

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.XWikiFullPage(*args, **kwargs)
```

Represents a full XWiki Page translation: this file does not contains properties but its whole content needs to be translated. More information on https://dev.xwiki.org/xwiki/bin/view/Community/XWiki%20Translations%20Formats/#HXWikiFullContentTranslation

UnitClass

alias of xwikiunit

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 removeunit (unit) Remove 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. 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

$\textbf{class} \texttt{ translate.storage.properties.XWikiPageProperties} \ (*args, **kwargs)$

unit iter()

Iterator over all the units in this store.

Represents an XWiki Page containing translation properties as described in https://dev.xwiki.org/xwiki/bin/view/Community/XWiki%20Translations%20Formats/#HXWikiPageProperties

UnitClass

alias of xwikiunit

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
     removeunit (unit)
          Remove 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.
     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.
translate.storage.properties.accesskeysuffixes = ('.accesskey', '.accessKey', '.akey')
     Accesskey Suffixes: entries with this suffix may be combined with labels ending in labelsuffixes into
     accelerator notation
class translate.storage.properties.gwtfile(*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.

${\tt remove_unit_from_index}\,(\mathit{unit})$

Remove a unit from source and locaton indexes

```
removeunit (unit)
          Remove 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.
     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.
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

getids (filename=None)

return a list of unit ids

getprojectstyle()

Get the project type for this store.

${\tt getsourcelanguage} \; (\;)$

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()
           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
     removeunit (unit)
           Remove 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.
     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.properties.javautf16file(*args, **kwargs)
```

isempty()

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.

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
     removeunit (unit)
           Remove 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.
     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.

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

removeunit (unit)

Remove 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.
     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
removeunit (unit)
     Remove 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.
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.
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
removeunit (unit)
     Remove 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.
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.proppluralunit (source=", personality='java')

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 (key=None)

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('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.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.

classmethod get_missing_part()

Return the string representing a missing translation.

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 (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(['foo', 'bar', 'baz'])
>>> TranslationUnit.multistring_to_rich(target)
[<StringElem([<StringElem(['foo'])>])>,
    <StringElem([<StringElem(['bar'])>])>,
    <StringElem([<StringElem(['baz'])>])>]
```

removenotes (origin=None)

Remove all the translator's notes.

classmethod represents_missing(line)

The line represents a missing translation

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('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

```
classmethod strip_missing_part(line)
          Remove the missing prefix from the line.
     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.
```

```
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
     removeunit (unit)
           Remove 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.
     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.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.

Returns The default matching criterion for all the subclasses.

The matching criterion to use when merging on.

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
     removeunit(unit)
          Remove 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.
     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.xwikifile(*args, **kwargs)
     UnitClass
          alias of xwikiunit
     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

removeunit (unit)

Remove 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.

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.xwikiunit(source=", personality='xwiki')

Represents an XWiki translation unit. The difference with a propunit is twofold:

- 1. the dialect used is xwiki for simple quote escape handling
- 2. missing translations are output with a dedicated "### Missing: " prefix

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.

classmethod get_missing_part()

Return the string representing a missing translation.

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.

 $\textbf{merge} \ (other unit, overwrite = False, comments = True, authoritative = False)$

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.
     classmethod represents_missing(line)
          Return true if the line represents a missing translation
     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('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
     classmethod strip_missing_part(line)
          Remove the missing prefix from the line.
     unit iter()
          Iterator that only returns this unit.
pypo
Classes that hold units of Gettext .po files (pounit) or entire files (pofile).
class translate.storage.pypo.PoWrapper(width=77,
                                                                   replace_whitespace=False,
                                                                                                 ex-
                                                    pand_tabs=False, drop_whitespace=False)
     fill (text : string) \rightarrow string
          Reformat the single paragraph in 'text' to fit in lines of no more than 'self.width' columns, and return a
          new string containing the entire wrapped paragraph.
     wrap (text : string) \rightarrow [string]
          Reformat the single paragraph in 'text' so it fits in lines of no more than 'self.width' columns, and return
          a list of wrapped lines. Tabs in 'text' are expanded with string.expandtabs(), and all other whitespace
```

4.1. API 621

characters (including newline) are converted to space.

Escapes a line for po format. assumes no occurs in the line.

translate.storage.pypo.escapeforpo(line)

param line unescaped text

```
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.

Use 'Report-Msgid-Bug-To' entry
 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
removeunit (unit)
     Remove 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.
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 TranslationUnit.addlocation().

Warning: This method might be removed in future.

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 sourcecomments 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, parsed=None)

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.

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()

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('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 gmunit
     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.

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)

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

removeunit (unit)

Remove 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.

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 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:

```
>>> target = multistring(['foo', 'bar', 'baz'])
>>> TranslationUnit.multistring_to_rich(target)
[<StringElem([<StringElem(['foo'])>])>,
    <StringElem([<StringElem(['bar'])>])>,
    <StringElem([<StringElem(['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('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

removeunit (unit)

Remove 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.

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)

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 <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:

```
>>> target = multistring(['foo', 'bar', 'baz'])
>>> TranslationUnit.multistring_to_rich(target)
[<StringElem([<StringElem(['foo'])>])>,
    <StringElem([<StringElem(['bar'])>])>,
    <StringElem([<StringElem(['baz'])>])>]
```

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('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.
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.
translate.storage.rc.generate_dialog_caption_name(block_type, identifier)
     Return the name generated for a caption of a dialog.
translate.storage.rc.generate_dialog_control_name(block_type, block_id, control_type,
                                                                identifier)
     Return the name generated for a control of a dialog.
translate.storage.rc.generate_menu_pre_name(block_type, block_id)
     Return the pre-name generated for elements of a menu.
translate.storage.rc.generate_menuitem_name(pre_name, block_type, identifier)
     Return the name generated for a menuitem of a popup.
translate.storage.rc.generate_popup_caption_name(pre_name)
     Return the name generated for a caption of a popup.
```

rc

```
translate.storage.rc.generate_popup_pre_name (pre_name, caption) Return the pre-name generated for subelements of a popup.
```

Parameters

- **pre_name** The pre_name that already have the popup.
- **caption** The caption (whitout quotes) of the popup.

Returns The subelements pre-name based in the pre-name of the popup and its caption.

```
translate.storage.rc.generate_stringtable_name (identifier)
```

Return the name generated for a stringtable element.

```
translate.storage.rc.rc_statement()
```

Generate a RC statement parser that can be used to parse a RC file

Return type pyparsing.ParserElement

```
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_popup_units (pre_name, popup)

Transverses the popup tree making new units as needed.

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 removeunit (unit) Remove 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. 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.

4.1. API 641

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.
- **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()).

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.

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('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.

statistics

Module to provide statistics and related functionality.

```
class translate.storage.statistics.Statistics(sourcelanguage='en', target-
language='en', checkerstyle=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.
               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.
```

4.1. API 647

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 removeunit (unit) Remove 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. 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.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

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

addunit (unit)

```
removeunit (unit)
           Remove 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.
     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.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.

Simple detection based on BOM in case chardet is not available.

fallback_detection (text)

```
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
removeunit (unit)
     Remove 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.
require_index()
     make sure source index exists
```

4.1. API 651

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_unit_from_index(unit)
     Remove a unit from source and locaton indexes
removeunit (unit)
     Remove 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.
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)
```

4.1. API 653

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
     removeunit (unit)
           Remove 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.
     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)
           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 <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()
```

$\verb|classmethod| \verb|rich_to_multistring| (elem_list)$

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('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

removeunit (unit)

Remove 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.

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.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

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 <code>qetlocations()</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 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('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.

tiki

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. makeindex() Indexes the items in this store. At least .sourceindex should be useful.

The matching criterion to use when merging on.

Returns The default matching criterion for all the subclasses.

Return type string

parse(input)

merge_on

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

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

removeunit (unit)

Remove 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.

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:

```
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('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)
     with_traceback()
          Exception.with_traceback(tb) - set self.__traceback__ to tb and return self.
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
```

4.1. API 669

tmx

addheader() Method to be overridden to initialise headers, etc. addsourceunit (source) Adds and returns a new unit with the given string as first entry. 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.

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 removeunit (unit) Remove 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. 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)

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 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.

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('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
<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
</TrU>
translate.storage.trados.TRADOS_TIMEFORMAT = '%d%m%Y, %H:%M:%S'
     Time format used by Trados .txt
translate.storage.trados.RTF_ESCAPES = {'\\-': '\xad', '\\_': '-', '\\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 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.

$\verb|istranslatable| ()$

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('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.
          Return True if the object doesn't contain any translation units.
     makeindex()
```

4.1. API 679

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.

remove_unit_from_index(unit)

Remove a unit from source and locaton indexes

removeunit (unit)

Remove 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.

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.

ts2

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, comment=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)
           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
```

4.1. API 681

Return type String

Get the target language for this .ts file.

Returns ISO code e.g. af, fr, pt BR

gettargetlanguage()

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

removeunit (unit)

Remove 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.

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)

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 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()

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.

 $\textbf{merge} \ (other unit, overwrite = False, comments = True, authoritative = False)$

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 ${\tt namespaced}$ ("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('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 = {'obsolete': -100, 'unfinished': 30, '': 100, None:
          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

ts

txt

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

removeunit (unit)

Remove 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.

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)

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 (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('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 Seperated 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.

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

removeunit (unit)

Remove 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.

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='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>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('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.

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 WordfastTMFile and WordfastUnit 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

```
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 implemenation.
translate.storage.wordfast.TAB UTF16 = b'\x00\t'
     The tab t character as it would appear in UTF-16 encoding
translate.storage.wordfast.WF_ESCAPE_MAP = (("&'26;", '&'), ("&'82;", ','), ("&'85;", '...
     Mapping of Wordfast &'XX; escapes to correct Unicode characters
translate.storage.wordfast.WF_FIELDNAMES = ['date', 'user', 'reuse', 'src-lang', 'source',
     Field names for a Wordfast TU
translate.storage.wordfast.WF_FIELDNAMES_HEADER = ['date', 'userlist', 'tucount', 'src-land
     Field names for the Wordfast header
translate.storage.wordfast.WF_FIELDNAMES_HEADER_DEFAULTS = { 'attr1list': '', 'attr2list'
     Default or minimum header entries for a Wordfast file
translate.storage.wordfast.WF TIMEFORMAT = '%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.
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.
```

Mac version. This might cause errors in future. Functions allow for <_wf_to_char> and back to

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
removeunit (unit)
     Remove 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.
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 699

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.

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(['foo', 'bar', 'baz'])
>>> TranslationUnit.multistring_to_rich(target)
[<StringElem([<StringElem(['foo'])>])>,
    <StringElem([<StringElem(['bar'])>])>,
    <StringElem([<StringElem(['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('foo bar')
```

setcontext (context)

Set the message context

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()

Iterator that only returns this unit.

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 $C \le n < MAX$.

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)
     with_traceback()
          Exception.with_traceback(tb) - set self.__traceback__ to tb and return self.
exception translate.storage.workflow.NoInitialStateError
     with traceback()
          Exception.with_traceback(tb) - set self.__traceback__ to tb and return self.
class translate.storage.workflow.StateEnum
     Only contains the constants for default states.
exception translate.storage.workflow.StateNotInWorkflowError(state)
     with traceback()
          Exception.with_traceback(tb) - set self.__traceback__ to tb and return self.
exception translate.storage.workflow.TransitionError
     with_traceback()
          Exception.with_traceback(tb) - set self.__traceback__ to tb and return self.
exception translate.storage.workflow.WorkflowError
     with_traceback()
          Exception.with_traceback(tb) – set self.__traceback__ to tb and return self.
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)
          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=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 705

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.

removeunit (unit)

Remove 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.

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

$\verb|setfilename|| (filenode, filename)|$

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.

${\tt addalttrans}\ (\textit{txt}, \textit{origin} = None, \textit{lang} = None, \textit{sourcetxt} = None, \textit{matchquality} = None)$

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

$\verb|isreview|()$

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.

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)

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

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.

```
translate.storage.xml_extract.extract.build_idml_store(odf_file, store, parse_state, store_adder=None)

Build a store for the given IDML file.
```

```
translate.storage.xml\_extract.extract.build\_store (odf\_file, store, parse\_state, store\_adder=None) \\ Build a store for the given XML file.
```

```
\verb|translate.storage.xml_extract.extract.make_postore_adder| (|store|, id_maker|, file-postore_adder)| |storage.xml_extract.extract.make_postore_adder| |storage.xml_extract.extract.make_postore_adder| |storage.xml_extract.extract.make_postore_adder| |storage.xml_extract.extract.make_postore_adder| |storage.xml_extract.extract.make_postore_adder| |storage.xml_extract.extract.make_postore_adder| |storage.xml_extract.make_postore_adder| |storage
```

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)

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)
```

(continues on next page)

(continued from previous page)

```
| `- ('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>

 then the breadcrumb will keep track of the number of times it sees

 Thus

```
>>> xb.start_tag('bar')
>>> xb.xpath
foo[0]/bar[0]
>>> xb.end_tag()
>>> xb.xpath
foo[0]
>>> xb.start_tag('bar')
>>> xb.xpath
foo[0]/bar[1]
```

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.

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

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.

${\tt getoutputoptions}\ (options, input path, template path)$

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.

${\tt getusageman}\ (option)$

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 or iterable) - 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.

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. \textbf{convertpo} (input file, output file, temp late file, format=None, rewrite style=None, ignore option=None, preserve place-holders=None)
```

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.

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

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 or iterable) - 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)
     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.
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 us-
age instructions.
translate.tools.pomerge.mergestores(store1, store2, mergeblanks, mergefuzzy, mergecom-
                                                 ments)
     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
          Return type Boolean
```

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.

```
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

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

```
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 or iterable) - 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.

recurseinputfilelist(options)

 $See: \ http://docs.translatehouse.org/projects/translate-toolkit/en/latest/commands/posegment.html \ for \ examples \ and \ usage \ instructions.$

```
translate. tools.posegment. \textbf{segmentfile} (inputfile, outputfile, template file, source-language='en', target language=None, stripspaces=True, only a ligned=False) \\ reads in input file, segments it then, writes to output file
```

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.
```

```
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.

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.

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)

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 or iterable) - 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 translate.tools.pretranslate.match fuzzy(input unit, matchers)

usage instructions.

```
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,
                                                                ers=None.
                                                                                  mark reused=False,
                                                                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
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)
```

pypo2phppo

Convert Python format .po files to PHP format .po files.

translate.tools.pydiff.main()
 main program for pydiff

writes the actual diff to the given file

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, 263
                                           translate.convert.po2xliff, 266
translate.convert, 236
                                           translate.convert.po2yaml, 266
translate.convert.accesskey, 236
                                           translate.convert.pot2po, 267
translate.convert.convert, 237
                                           translate.convert.prop2mozfunny, 267
translate.convert.csv2po, 244
                                           translate.convert.prop2po, 268
translate.convert.csv2tbx, 245
                                           translate.convert.rc2po, 269
translate.convert.dtd2po, 245
                                           translate.convert.resx2po, 269
translate.convert.factory, 246
                                           translate.convert.sub2po, 269
translate.convert.html2po,246
                                           translate.convert.symb2po, 270
translate.convert.ical2po, 246
                                           translate.convert.tiki2po, 270
translate.convert.ini2po, 247
                                           translate.convert.ts2po, 270
translate.convert.json2po,247
                                           translate.convert.txt2po, 271
translate.convert.moz2po, 248
                                           translate.convert.web2py2po,271
translate.convert.mozfunny2prop, 248
                                           translate.convert.xliff2odf,271
translate.convert.mozlang2po, 248
                                           translate.convert.xliff2oo,272
translate.convert.odf2xliff, 249
                                           translate.convert.xliff2po, 272
translate.convert.oo2po, 249
                                           translate.convert.yaml2po, 272
translate.convert.oo2xliff, 249
                                           translate.filters.272
translate.convert.php2po, 250
                                           translate.filters.autocorrect, 273
translate.convert.po2csv, 250
                                           translate.filters.checks, 273
translate.convert.po2dtd, 250
                                           translate.filters.decoration, 351
translate.convert.po2html, 251
                                           translate.filters.helpers,352
translate.convert.po2ical, 251
                                           translate.filters.pofilter, 352
translate.convert.po2ini, 251
                                           translate.filters.prefilters, 355
translate.convert.po2json,252
                                           translate.filters.spelling, 356
translate.convert.po2moz, 253
                                           translate.lang, 356
translate.convert.po2mozlang, 252
                                           translate.lang.af, 357
translate.convert.po2oo, 256
                                           translate.lang.am, 358
translate.convert.po2php, 256
                                           translate.lang.ar, 358
translate.convert.po2prop, 256
                                           translate.lang.bn, 359
translate.convert.po2rc, 257
                                           translate.lang.code or, 360
translate.convert.po2resx, 257
                                           translate.lang.common, 361
translate.convert.po2sub, 257
                                           translate.lang.data, 364
translate.convert.po2symb, 257
                                           translate.lang.de, 365
translate.convert.po2tiki, 257
                                           translate.lang.el, 366
translate.convert.po2tmx, 258
                                           translate.lang.es, 367
translate.convert.po2ts, 262
                                           translate.lang.fa, 368
translate.convert.po2txt, 262
                                           translate.lang.factory, 368
translate.convert.po2web2py, 262
                                           translate.lang.fi, 369
```

```
translate.lang.fr, 369
                                           translate.storage.ical, 441
translate.lang.gu, 370
                                           translate.storage.ini,446
translate.lang.he, 371
                                           translate.storage.jsonl10n,451
translate.lang.hi,372
                                           translate.storage.lisa,482
translate.lang.hy, 373
                                           translate.storage.mo, 487
translate.lang.identify, 373
                                           translate.storage.mozilla lang,493
translate.lang.ja,373
                                           translate.storage.odf io,498
translate.lang.km, 374
                                           translate.storage.odf shared, 498
translate.lang.kn, 375
                                           translate.storage.omegat,498
translate.lang.ko, 376
                                           translate.storage.oo, 505
translate.lang.ml, 377
                                           translate.storage.php, 557
translate.lang.mr, 377
                                           translate.storage.placeables, 508
translate.lang.ne, 378
                                          translate.storage.placeables.base, 508
translate.lang.ngram, 379
                                           translate.storage.placeables.general,
translate.lang.pa, 379
                                                  523
translate.lang.poedit, 380
                                           translate.storage.placeables.interfaces,
translate.lang.si,380
translate.lang.st, 381
                                           translate.storage.placeables.lisa,536
translate.lang.sv, 382
                                           translate.storage.placeables.parse, 536
translate.lang.ta, 383
                                           translate.storage.placeables.strelem,
translate.lang.te, 384
                                                  537
translate.lang.team, 383
                                           translate.storage.placeables.terminology,
translate.lang.th, 384
                                                  539
translate.lang.ug.385
                                           translate.storage.placeables.xliff,541
translate.lang.ur, 386
                                           translate.storage.po, 575
translate.lang.vi, 387
                                           translate.storage.pocommon, 568
translate.lang.zh, 388
                                           translate.storage.poheader, 574
translate.misc, 388
                                           translate.storage.poparser, 575
translate.misc.dictutils, 388
                                           translate.storage.poxliff, 576
translate.misc.file_discovery, 389
                                           translate.storage.project, 583
translate.misc.multistring, 389
                                           translate.storage.projstore, 584
translate.misc.optrecurse, 393
                                           translate.storage.properties, 585
translate.misc.ourdom, 397
                                           translate.storage.pypo, 621
translate.misc.progressbar, 398
                                           translate.storage.qm, 628
translate.misc.quote, 399
                                           translate.storage.gph, 633
translate.misc.wsgi, 400
                                           translate.storage.rc,639
translate.misc.xml helpers, 400
                                           translate.storage.statistics, 645
translate.search, 401
                                           translate.storage.statsdb, 646
translate.search.lshtein,401
                                           translate.storage.subtitles,646
translate.search.match, 401
                                           translate.storage.symbian, 659
translate.search.terminology, 403
                                           translate.storage.tbx,659
translate.services, 403
                                           translate.storage.tiki,664
translate.services.tmserver.403
                                           translate.storage.tmdb,669
translate.storage, 403
                                           translate.storage.tmx, 669
translate.storage._factory_classes,432
                                           translate.storage.trados,675
translate.storage.base, 403
                                           translate.storage.ts,686
translate.storage.benchmark,414
                                           translate.storage.ts2,680
translate.storage.bundleprojstore, 414
                                           translate.storage.txt,686
translate.storage.catkeys, 415
                                           translate.storage.utx,691
translate.storage.csvl10n,421
                                           translate.storage.wordfast,697
translate.storage.directory, 426
                                           translate.storage.workflow, 703
translate.storage.dtd, 426
                                           translate.storage.xliff, 704
translate.storage.factory, 432
                                           translate.storage.xml_extract,711
translate.storage.html, 433
```

```
translate.storage.xml_extract.extract,
       711
translate.storage.xml_extract.generate,
      711
translate.storage.xml_extract.misc,712
translate.storage.xml_extract.unit_tree,
translate.storage.xml_extract.xpath_breadcrumb,
      713
translate.storage.xml_name,713
translate.storage.zip, 714
translate.tools,714
translate.tools.build_tmdb,714
translate.tools.phppo2pypo,714
translate.tools.poclean,715
translate.tools.pocompile,715
translate.tools.poconflicts, 715
translate.tools.pocount,718
translate.tools.podebug, 719
translate.tools.pogrep,719
translate.tools.pomerge, 722
translate.tools.porestructure, 722
translate.tools.posegment,725
translate.tools.poswap, 726
translate.tools.poterminology, 726
translate.tools.pretranslate,729
translate.tools.pydiff,730
translate.tools.pypo2phppo,730
```

Α		accesskeysuffixes (in module trans-
accelerators()	(trans-	late.storage.dtd), 427
late. filters. checks. CCL icense Checker	method),	accesskeysuffixes (in module trans-
273		late.storage.properties), 595
accelerators()	(trans-	acronyms () (translate.filters.checks.CCLicenseChecker method), 273
v A	method),	acronyms () (translate.filters.checks.DrupalChecker
279		method), 279
accelerators()	(trans- method),	acronyms() (translate.filters.checks.GnomeChecker
late.filters.checks.GnomeChecker 285	тетоа),	method), 285
accelerators()(translate.filters.checks.IO	SChecker	acronyms() (translate.filters.checks.IOSChecker
method), 291		method), 291
accelerators()	(trans-	acronyms() (translate.filters.checks.KdeChecker
· ·	method),	method), 296
296		acronyms() (translate.filters.checks.L20nChecker method), 302
accelerators()	(trans-	acronyms () (translate.filters.checks.LibreOfficeChecker
· ·	method),	method), 308
302	(trans-	acronyms() (translate.filters.checks.MinimalChecker
<pre>accelerators() late.filters.checks.LibreOfficeChecker</pre>	(method), 314
308	memou),	acronyms() (translate.filters.checks.MozillaChecker
accelerators()	(trans-	method), 319
• • • • • • • • • • • • • • • • • • • •	method),	acronyms () (translate.filters.checks.OpenOfficeChecker
314	,,	method), 325
accelerators()	(trans-	acronyms() (translate.filters.checks.ReducedChecker
late.filters.checks.Mozilla Checker	method),	method), 331
319		acronyms () (translate.filters.checks.StandardChecker
accelerators()	(trans-	method), 337 acronyms() (translate.filters.checks.TermChecker
late.filters.checks.OpenOfficeChecker	method),	acronyms () (translate.filters.checks.TermChecker method), 344
325	(tuans	add_duplicates_option() (trans-
<pre>accelerators() late.filters.checks.ReducedChecker</pre>	(trans- method),	late.convert.convert.ArchiveConvertOptionParser
331	memou),	method), 237
accelerators()	(trans-	add_duplicates_option() (trans-
	method),	late.convert.convert.ConvertOptionParser
337	,,	method), 241
accelerators()	(trans-	add_duplicates_option() (trans-
late.filters.checks.TermChecker	method),	late.convert.po2moz.MozConvertOptionParser
344		method), 253
		add_duplicates_option() (trans-

late.convert.po2tmx.TmxOptionParser method), 258		late.misc.optrecurse. method), 394	RecursiveOptionParser
add_duplicates_option()	(trans-	dd_option()	(trans-
late.convert.po2wordfast.WfOptionParsomethod), 263	er	late.tools.poconflicts method), 715	.ConflictOptionParser
add_fuzzy_option()	(trans-	dd_option()	(trans-
late.convert.convert.ArchiveConvertOpt method), 237	ionParser	late.tools.pogrep.Gre 719	epOptionParser method),
add_fuzzy_option()	(trans-	dd_option()	(trans-
late.convert.convert.ConvertOptionPars method), 241	er	late.tools.porestructi method), 723	ure.SplitOptionParser
add_fuzzy_option()	(trans-	dd_option()	(trans-
late.convert.po2moz.MozConvertOption	Parser	late.tools.poterminol	logy.TerminologyOptionParser
method), 253		method), 726	
<pre>add_fuzzy_option()</pre>	(trans-	dd_popup_units()	(translate.storage.rc.rcfile
late.convert.po2tmx.TmxOptionParser		method), 640	
method), 258		dd_remove_untranslat	ted_option() (trans-
<pre>add_fuzzy_option()</pre>	(trans-	late.convert.convert.	ArchiveConvertOptionParser
late.convert.po2wordfast.WfOptionParso	er	method), 237	
method), 263		dd_remove_untranslat	ted_option() (trans-
<pre>add_multifile_option()</pre>	(trans-	late.convert.convert.	ConvertOptionParser
late.convert.convert.ArchiveConvertOpt	ionParser	method), 241	
method), 237		dd_remove_untranslat	ted_option() (trans-
<pre>add_multifile_option()</pre>	(trans-	late.convert.po2moz.	MozConvertOptionParser
late.convert.convert.ConvertOptionPars	er	method), 253	
method), 241		dd_remove_untranslat	ted_option() (trans-
<pre>add_multifile_option()</pre>	(trans-	late.convert.po2tmx.	TmxOptionParser
late.convert.po2moz.MozConvertOption	Parser	method), 258	
method), 253		dd_remove_untranslat	ted_option() (trans-
<pre>add_multifile_option()</pre>	(trans-	late.convert.po2word method), 263	lfast.WfOptionParser
method), 258		dd_source() (trans	late.storage.project.Project
add_multifile_option()	(trans-	method), 583	
late.convert.po2wordfast.WfOptionPars	er	dd_source_convert()	(trans-
method), 263		late.storage.project.l	Project method), 583
<pre>add_option()</pre>	(trans-	dd_spreadsheet_escap	pes() (trans-
late.convert.convert.ArchiveConvertOpt	ionParser	late.storage.csvl10n.	csvunit method), 422
method), 237		dd_threshold_option	() (trans-
<pre>add_option()</pre>	(trans-	late.convert.convert.	ArchiveConvertOptionParser
late.convert.convert.ConvertOptionPars	er	method), 237	
method), 241		dd_threshold_option	() (trans-
add_option()	(trans-	late.convert.convert.	ConvertOptionParser
late.convert.po2moz.MozConvertOption	Parser	method), 241	
method), 253		dd_threshold_option	() (trans-
add_option()	(trans-	late.convert.po2moz.	MozConvertOptionParser
late.convert.po2tmx.TmxOptionParser		method), 253	
method), 258		dd_threshold_option	() (trans-
add_option()	(trans-	late.convert.po2tmx.	TmxOptionParser
late.convert.po2wordfast.WfOptionPars	er	method), 258	
method), 263		dd_threshold_option	() (trans-
add_option()	(trans-		lfast.WfOptionParser
late.filters.pofilter.FilterOptionParser m	nethod),	method), 263	
352		dd_unit_to_index()	(trans-
<pre>add_option()</pre>	(trans-	late.storage.base.Dic	ctStore method), 403

add_u	<pre>nit_to_index() late.storage.base.TranslationStore</pre>	(trans- method),	<pre>add_unit_to_index()</pre>	
	408		add_unit_to_index()	(trans-
add_u	nit_to_index()	(trans-	late.storage.poxliff.PoXliffFile method	
	late.storage.catkeys.CatkeysFile	method),	add_unit_to_index()	(trans-
299	415	(trans-	late.storage.properties.gwtfile method), 393 (trans-
auu_u	<pre>nit_to_index() late.storage.csvl10n.csvfile method), 4</pre>		<pre>add_unit_to_index() late.storage.properties.javafile</pre>	method),
add 11	nit_to_index() (translate.storage.		598	memou),
aaa_a	method), 427	aia.aiajiic	add_unit_to_index()	(trans-
add 11	nit_to_index()	(trans-		method),
a a a a_a	late.storage.html.htmlfile method), 430		600	,
add u	nit_to_index()		add_unit_to_index()	(trans-
	late.storage.html.POHTMLParser			method),
	433	,,	601	,,
add_u	nit_to_index()	(trans-	add_unit_to_index()	(trans-
	late.storage.ical.icalfile method), 441		late.storage.properties.joomlafile	method),
add_u	<pre>nit_to_index() (translate.storage</pre>	ini.inifile	603	
	method), 446		<pre>add_unit_to_index()</pre>	(trans-
add_u	nit_to_index()	(trans-	~	method),
	~ ·	method),	605	
	452		<pre>add_unit_to_index()</pre>	(trans-
add_u	nit_to_index()	(trans-		method),
	late.storage.jsonl10n.GoI18NJsonFile method), 457	?	613 add_unit_to_index()	(trans-
add_u	nit_to_index()	(trans-	late.storage.properties.stringsutf8file	method),
		method),	615	
	462		add_unit_to_index()	(trans-
add_u	nit_to_index()	(trans-	~ · ·	method),
	late.storage.jsonl10n.JsonFile method		616	
add_u	nit_to_index()	(trans-	add_unit_to_index()	(trans-
	late.storage.jsonl10n.JsonNestedFile 469	metnoa),	late.storage.properties.XWikiFullPage	
299	nit_to_index()	(trans	<pre>method), 592 add_unit_to_index()</pre>	(trans-
auu_u	late.storage.jsonl10n.WebExtensionJs		late.storage.properties.XWikiPageProp	`
	method), 477	oni ne	method), 594	<i>jernes</i>
add u	nit_to_index()	(trans-	add_unit_to_index()	(trans-
	late.storage.lisa.LISAfile method), 482		late.storage.pypo.pofile method), 622	(
	nit_to_index() (translate.storage.		add_unit_to_index() (translate.storage.d	gm.qmfile
	method), 487	v	method), 629	1 1 0
add_u	nit_to_index()	(trans-	add_unit_to_index()	(trans-
	late.storage.mozilla_lang.LangStore	method),	late.storage.qph.QphFile method), 634	4
	493		<pre>add_unit_to_index() (translate.storag</pre>	e.rc.rcfile
add_u	nit_to_index()	(trans-	method), 640	
	late.storage.omegat.OmegaTFile	method),	add_unit_to_index()	(trans-
	498		late. storage. subtitles. Adv SubStation Algebras and the state of t	phaFile
add_u	nit_to_index()	(trans-	method), 647	
	late.storage.omegat.OmegaTFileTab	method),	<pre>add_unit_to_index()</pre>	(trans-
	500		o e e e e e e e e e e e e e e e e e e e	method),
add_u	nit_to_index()	(trans-	648	,
	late.storage.php.LaravelPHPFile	method),	add_unit_to_index()	(trans-
	558	(:	•	method),
add_u	nit_to_index()	(trans-	650	(4-
	late.storage.php.phpfile method), 563		<pre>add_unit_to_index()</pre>	(trans-

late.storage.subtitles.SubStationAlphaFile method), 652	adderror() (translate.storage.jsonl10n.WebExtensionJsonUnit method), 478
add_unit_to_index() (trans- late.storage.subtitles.SubtitleFile method),	adderror() (translate.storage.lisa.LISAunit method), 484
654 add_unit_to_index() (translate.storage.tbx.tbxfile	adderror() (translate.storage.mo.mounit method), 490
method), 659 add_unit_to_index() (trans-	adderror() (translate.storage.mozilla_lang.LangUnit method), 495
late.storage.tiki.TikiStore method), 664 add_unit_to_index() (trans-	adderror() (translate.storage.omegat.OmegaTUnit method), 502
late.storage.tmx.tmxfile method), 669 add_unit_to_index() (trans-	adderror() (translate.storage.php.LaravelPHPUnit method), 560
late.storage.trados.TradosTxtTmFile method), 679	adderror() (translate.storage.php.phpunit method), 565
add_unit_to_index() (translate.storage.ts2.tsfile method), 681	adderror() (translate.storage.pocommon.pounit method), 571
add_unit_to_index() (trans-	adderror() (translate.storage.poxliff.PoXliffUnit
late.storage.txt.TxtFile method), 686	method), 579
add_unit_to_index() (trans- late.storage.utx.UtxFile method), 692	adderror() (translate.storage.properties.proppluralunit method), 607
<pre>add_unit_to_index() (trans- late.storage.wordfast.WordfastTMFile method),</pre>	adderror() (translate.storage.properties.propunit method), 610
698	adderror() (translate.storage.properties.xwikiunit
add_unit_to_index() (trans-	method), 618
late.storage.xliff.xlifffile method), 704	adderror() (translate.storage.pypo.pounit method),
addalttrans() (translate.storage.poxliff.PoXliffUnit method), 579	625 adderror() (translate.storage.qm.qmunit method),
addalttrans() (translate.storage.xliff.xliffunit	630
method), 707 adderror() (translate.storage.base.DictUnit method),	adderror() (translate.storage.qph.QphUnit method), 636
405	adderror() (translate.storage.rc.rcunit method), 642
adderror() (translate.storage.base.TranslationUnit method), 411	adderror() (translate.storage.subtitles.SubtitleUnit method), 655
adderror() (translate.storage.catkeys.CatkeysUnit method), 417	adderror() (translate.storage.tbx.tbxunit method), 661
adderror() (translate.storage.csvl10n.csvunit method), 422	adderror() (translate.storage.tiki.TikiUnit method), 666
adderror() (translate.storage.dtd.dtdunit method), 429	adderror() (translate.storage.tmx.tmxunit method), 671
adderror() (translate.storage.html.htmlunit method),	adderror() (translate.storage.trados.TradosUnit
438	method), 676
adderror() (translate.storage.ical.icalunit method), 443	adderror() (translate.storage.ts2.tsunit method), 683 adderror() (translate.storage.txt.TxtUnit method),
adderror() (translate.storage.ini.iniunit method), 448 adderror() (translate.storage.jsonl10n.ARBJsonUnit method), 454	688 adderror() (translate.storage.utx.UtxUnit method), 694
adderror() (translate.storage.jsonl10n.GoI18NJsonUni method), 459	
	adderror() (translate.storage.xliff.xliffunit method),
adderror() (translate.storage.jsonl10n.JsonNestedUnit method), 470	
adderror() (translate.storage.jsonl10n.JsonUnit method), 474	addheader() (translate.storage.poxliff.PoXliffFile method), 576

addheader() (translate.storage.qph.QphFile method), 634	late.storage.omegat.OmegaTUnit method), 502
addheader() (translate.storage.tbx.tbxfile method),	addlocation() (trans-
659	late.storage.php.LaravelPHPUnit method),
<pre>addheader() (translate.storage.tmx.tmxfile method),</pre>	560
669	addlocation() (translate.storage.php.phpunit
addheader() (translate.storage.ts2.tsfile method), 681	method), 565
addheader() (translate.storage.xliff.xlifffile method), 704	addlocation() (translate.storage.pocommon.pounit method), 571
addline() (translate.storage.oo.oofile method), 506	addlocation() (translate.storage.poxliff.PoXliffUnit
addline() (translate.storage.oo.oounit method), 507	method), 579
addlocation() (translate.storage.base.DictUnit	addlocation() (trans-
method), 405	late.storage.properties.proppluralunit method),
addlocation() (trans-	607
late.storage.base.TranslationUnit method),	addlocation() (trans-
411	late.storage.properties.propunit method),
addlocation() (trans-	610
late.storage.catkeys.CatkeysUnit method),	addlocation() (trans-
417	late.storage.properties.xwikiunit method),
addlocation() (translate.storage.csvl10n.csvunit	618
method), 423	addlocation() (translate.storage.pypo.pounit
addlocation() (translate.storage.dtd.dtdunit	method), 625
method), 429	addlocation() (translate.storage.qm.qmunit
addlocation() (translate.storage.html.htmlunit	method), 630
method), 438	addlocation() (translate.storage.qph.QphUnit
addlocation() (translate.storage.ical.icalunit	method), 636
method), 443	addlocation() (translate.storage.rc.rcunit method),
addlocation() (translate.storage.ini.iniunit method),	642
448	addlocation() (trans-
addlocation() (trans-	late.storage.subtitles.SubtitleUnit method),
late.storage.jsonl10n.ARBJsonUnit method), 454	656
	addlocation() (translate.storage.tbx.tbxunit
addlocation() (trans-	<pre>method), 661 addlocation() (translate.storage.tiki.TikiUnit</pre>
late.storage.jsonl10n.GoI18NJsonUnit method), 459	method), 666
addlocation() (trans-	addlocation() (translate.storage.tmx.tmxunit
late.storage.jsonl10n.I18NextUnit method),	method), 671
464	addlocation() (translate.storage.trados.TradosUnit
addlocation() (trans-	method), 676
late.storage.jsonl10n.JsonNestedUnit method),	addlocation() (translate.storage.ts2.tsunit method),
471	683
addlocation() (translate.storage.jsonl10n.JsonUnit	addlocation() (translate.storage.txt.TxtUnit
method), 474	method), 688
addlocation() (trans-	addlocation() (translate.storage.utx.UtxUnit
late.storage.jsonl10n.WebExtensionJsonUnit	method), 694
method), 479	addlocation() (trans-
addlocation() (translate.storage.lisa.LISAunit	late.storage.wordfast.WordfastUnit method),
method), 484	700
addlocation() (translate.storage.mo.mounit method), 490	addlocation() (translate.storage.xliff.xliffunit method), 707
addlocation() (trans-	addlocations() (translate.storage.base.DictUnit
late.storage.mozilla_lang.LangUnit method),	method), 405
495	addlocations() (trans-
addlocation() (trans-	late.storage.base.TranslationUnit method),

411	$late. storage. properties. propplural unit\ method),$
addlocations() (trans-	607
late.storage.catkeys.CatkeysUnit method),	addlocations() (trans-
417	late.storage.properties.propunit method),
addlocations() (translate.storage.csvl10n.csvunit	610
method), 423	addlocations() (trans-
addlocations() (translate.storage.dtd.dtdunit method), 429	late.storage.properties.xwikiunit method), 618
addlocations() (translate.storage.html.htmlunit	addlocations() (translate.storage.pypo.pounit
method), 438	method), 625
addlocations() (translate.storage.ical.icalunit method), 443	addlocations() (translate.storage.qm.qmunit method), 631
addlocations() (translate.storage.ini.iniunit	addlocations() (translate.storage.qph.QphUnit
method), 448	method), 636
addlocations() (trans-	addlocations() (translate.storage.rc.rcunit
late.storage.jsonl10n.ARBJsonUnit method),	method), 642
454	addlocations() (trans-
addlocations() (trans-	$late. storage. subtitle s. Subtitle Unit \\ method),$
late.storage.jsonl10n.GoI18NJsonUnit	656
method), 459	addlocations() (translate.storage.tbx.tbxunit
addlocations() (trans-	method), 661
late.storage.jsonl10n.I18NextUnit method), 464	
addlocations() (trans-	<pre>method), 666 addlocations() (translate.storage.tmx.tmxunit</pre>
late.storage.jsonl10n.JsonNestedUnit method),	method), 672
471	addlocations() (trans-
addlocations() (trans-	late.storage.trados.TradosUnit method),
late.storage.jsonl10n.JsonUnit method),	676
474	addlocations() (translate.storage.ts2.tsunit
addlocations() (trans-	method), 683
late.storage.jsonl 10 n. WebExtension Json Unit	addlocations() (translate.storage.txt.TxtUnit
method), 479	method), 688
addlocations() (translate.storage.lisa.LISAunit method), 484	addlocations() (translate.storage.utx.UtxUnit method), 694
addlocations() (translate.storage.mo.mounit	**
method), 490	late.storage.wordfast.WordfastUnit method),
addlocations() (trans-	700
late.storage.mozilla_lang.LangUnit method),	addlocations() (translate.storage.xliff.xliffunit
495	method), 707
addlocations() (trans-	<pre>addnote() (translate.storage.base.DictUnit method),</pre>
late.storage.omegat.OmegaTUnit method),	406
502	addnote() (translate.storage.base.TranslationUnit
addlocations() (trans-	method), 411
late.storage.php.LaravelPHPUnit method),	
addlocations() (translate.storage.php.phpunit	method), 418 addnote() (translate.storage.csvl10n.csvunit method),
method), 565	423
addlocations() (trans-	
late.storage.pocommon.pounit method), 571	addnote() (translate.storage.html.htmlunit method), 438
addlocations() (trans-	addnote() (translate.storage.ical.icalunit method),
$late. storage. pox liff. PoX liff Unit \\ method),$	443
580	addnote() (translate.storage.ini.iniunit method), 448
addlocations() (trans-	addnote() (translate.storage.jsonl10n.ARBJsonUnit

method), 454	707
addnote() (translate.storage.jsonl10n.GoI18NJsonUnit method), 459	addplural() (translate.storage.poxliff.PoXliffFile method), 576
addnote() (translate.storage.jsonl10n.I18NextUnit method), 464	addsourceunit() (translate.storage.base.DictStore method), 403
addnote() (translate.storage.jsonl10n.JsonNestedUnit	addsourceunit() (trans-
method), 471	late.storage.base.TranslationStore method),
addnote() (translate.storage.jsonl10n.JsonUnit	409
method), 474	addsourceunit() (trans-
addnote() (translate.storage.jsonl10n.WebExtensionJsonmethod), 479	nUnit late.storage.catkeys.CatkeysFile method), 415
addnote() (translate.storage.lisa.LISAunit method), 484	addsourceunit() (translate.storage.csvl10n.csvfile method), 421
addnote() (translate.storage.mo.mounit method), 491	addsourceunit() (translate.storage.dtd.dtdfile
<pre>addnote() (translate.storage.mozilla_lang.LangUnit</pre>	method), 427
method), 495	addsourceunit() (translate.storage.html.htmlfile
addnote() (translate.storage.omegat.OmegaTUnit	method), 436
method), 502	addsourceunit() (trans-
addnote() (translate.storage.php.LaravelPHPUnit method), 560	late.storage.html.POHTMLParser method), 433
addnote() (translate.storage.php.phpunit method), 565	addsourceunit() (translate.storage.ical.icalfile method), 441
addnote() (translate.storage.pocommon.pounit	addsourceunit() (translate.storage.ini.inifile
method), 571	method), 446
addnote() (translate.storage.poxliff.PoXliffUnit	addsourceunit() (trans-
method), 580	$late. storage. js on 110 n. ARB Js on File \\ method),$
$\verb"addnote" () (\textit{translate.storage.properties.propplural unit}$	452
method), 607	addsourceunit() (trans-
addnote() (translate.storage.properties.propunit method), 610	late.storage.jsonl10n.GoI18NJsonFile method), 457
addnote() (translate.storage.properties.xwikiunit	addsourceunit() (trans-
method), 619	late.storage.jsonl10n.I18NextFile method),
addnote() (translate.storage.pypo.pounit method),	462
625	addsourceunit() (trans-
addnote() (translate.storage.qm.qmunit method), 631	late.storage.jsonl10n.JsonFile method), 467
addnote() (translate.storage.qph.QphUnit method),	addsourceunit() (trans-
636	late.storage.jsonl10n.JsonNestedFile method),
addnote() (translate.storage.rc.rcunit method), 642	469
addnote() (translate.storage.subtitles.SubtitleUnit	addsourceunit() (trans-
method), 656	late.storage.jsonl10n.WebExtensionJsonFile
addnote() (translate.storage.tbx.tbxunit method), 661	method), 477
addnote() (translate.storage.tiki.TikiUnit method), 667	addsourceunit() (translate.storage.lisa.LISAfile method), 482
addnote() (translate.storage.tmx.tmxunit method), 672	addsourceunit() (translate.storage.mo.mofile method), 487
addnote() (translate.storage.trados.TradosUnit	addsourceunit() (trans-
method), 676	late.storage.mozilla_lang.LangStore method),
addnote() (translate.storage.ts2.tsunit method), 683	493
addnote() (translate.storage.txt.TxtUnit method), 689	addsourceunit() (trans-
addnote() (translate.storage.utx.UtxUnit method), 694	late.storage.omegat.OmegaTFile method), 499
addnote() (translate.storage.wordfast.WordfastUnit	addsourceunit() (trans-
method), 701	late. storage. omegat. OmegaTFile Tab method),
addnote() (translate.storage.xliff.xliffunit method),	500

addsourceunit()	(trans-	648
late.storage.php.LaravelPHPFile	method),	addsourceunit() (trans-
558		late.storage.subtitles.SubRipFile method),
addsourceunit() (translate.storage.	php.phpfile	650
method), 563		addsourceunit() (trans-
addsourceunit()	(trans-	late.storage.subtitles.SubStationAlphaFile
late.storage.pocommon.pofile metho	`	method), 652
addsourceunit()	(trans-	addsourceunit() (trans-
late.storage.poxliff.PoXliffFile metho	`	late.storage.subtitles.SubtitleFile method),
addsourceunit()	(trans-	654
late.storage.properties.gwtfile metho	`	addsourceunit() (translate.storage.tbx.tbxfile
addsourceunit()	(trans-	method), 659
late.storage.properties.javafile	method),	addsourceunit() (translate.storage.tiki.TikiStore
598	memoa),	
	(tuana	method), 665
addsourceunit()	(trans-	addsourceunit() (translate.storage.tmx.tmxfile
late.storage.properties.javautf16file	method),	method), 670
600		addsourceunit() (trans-
addsourceunit()	(trans-	late.storage.trados.TradosTxtTmFile method),
late.storage.properties.javautf8file	method),	679
601		addsourceunit() (translate.storage.ts2.tsfile
addsourceunit()	(trans-	method), 681
late.storage.properties.joomlafile	method),	addsourceunit() (translate.storage.txt.TxtFile
603		method), 687
addsourceunit()	(trans-	addsourceunit() (translate.storage.utx.UtxFile
late.storage.properties.propfile	method),	method), 692
605		addsourceunit() (trans-
addsourceunit()	(trans-	late.storage.wordfast.WordfastTMFile method),
late.storage.properties.stringsfile	method),	698
613		addsourceunit() (translate.storage.xliff.xlifffile
addsourceunit()	(trans-	method), 704
late.storage.properties.stringsutf8file	•	addtranslation() (translate.storage.tmx.tmxfile
615	,,	method), 670
addsourceunit()	(trans-	addunit() (translate.storage.base.DictStore method),
late.storage.properties.xwikifile	method),	403
616	,	addunit() (translate.storage.base.TranslationStore
addsourceunit()	(trans-	method), 409
late.storage.properties.XWikiFullPag		addunit() (translate.storage.catkeys.CatkeysFile
method), 592	50	method), 415
addsourceunit()	(trans-	addunit() (translate.storage.csvl10n.csvfile method),
late.storage.properties.XWikiPagePr	,	421
method), 594	opernes	
		addunit () (translate.storage.dtd.dtdfile method), 427
addsourceunit() (translate.storage.	руро.ројне	addunit() (translate.storage.html.htmlfile method),
method), 622	C1.	436
addsourceunit() (translate.storage	е.qт.qтлие	addunit() (translate.storage.html.POHTMLParser
method), 629	1 0 1 511	method), 433
addsourceunit() (translate.storage.q	ph.QphFile	addunit () (translate.storage.ical.icalfile method), 441
method), 634	2.	addunit () (translate.storage.ini.inifile method), 446
addsourceunit() (translate.store	age.rc.rcfile	addunit() (translate.storage.jsonl10n.ARBJsonFile
method), 640		method), 452
addsourceunit()	(trans-	addunit()(translate.storage.jsonl10n.GoI18NJsonFile
late. storage. subtitles. Adv Sub Station.	AlphaFile	method), 457
method), 647		addunit() (translate.storage.jsonl10n.118NextFile
addsourceunit()	(trans-	method), 462
late.storage.subtitles.MicroDVDFile	method),	addunit() (translate.storage.jsonl10n.JsonFile

method), 467	addunit()(translate.storage.subtitles.SubStationAlphaFile
addunit() (translate.storage.jsonl10n.JsonNestedFile	method), 652
method), 469	addunit() (translate.storage.subtitles.SubtitleFile
addunit() (translate.storage.jsonl10n.WebExtensionJso	
method), 477	addunit () (translate.storage.tbx.tbxfile method), 659
addunit() (translate.storage.lisa.LISAfile method), 482	addunit() (translate.storage.tiki.TikiStore method), 665
addunit () (translate.storage.mo.mofile method), 487	addunit() (translate.storage.tmx.tmxfile method), 670
addunit () (translate.storage.mozilla_lang.LangStore method), 493	addunit() (translate.storage.trados.TradosTxtTmFile method), 679
addunit() (translate.storage.omegat.OmegaTFile	addunit() (translate.storage.ts2.tsfile method), 681
method), 499	addunit () (translate.storage.txt.TxtFile method), 687
addunit () (translate.storage.omegat.OmegaTFileTab	addunit () (translate.storage.utx.UtxFile method), 692
method), 500 addunit() (translate.storage.php.LaravelPHPFile	addunit () (translate.storage.wordfast.WordfastTMFile method), 698
method), 558	addunit () (translate.storage.xliff.xlifffile method), 704
addunit () (translate.storage.php.phpfile method), 563	AdvSubStationAlphaFile (class in trans-
addunit() (translate.storage.pocommon.pofile	late.storage.subtitles), 646
method), 568	af (class in translate.lang.af), 357
addunit() (translate.storage.poxliff.PoXliffFile	AltAttrPlaceable (class in trans-
method), 576 addunit () (translate.storage.properties.gwtfile	<pre>late.storage.placeables.general), 523 alter_length() (translate.lang.af.af class method),</pre>
method), 595	357
addunit () (translate.storage.properties.javafile method), 598	alter_length() (translate.lang.am.am class method), 358
addunit () (translate.storage.properties.javautf16file method), 600	<pre>alter_length() (translate.lang.ar.ar class method),</pre>
addunit () (translate.storage.properties.javautf8file method), 601	alter_length() (translate.lang.bn.bn class method), 359
addunit () (translate.storage.properties.joomlafile method), 603	<pre>alter_length() (translate.lang.code_or.code_or</pre>
addunit () (translate.storage.properties.propfile method), 605	alter_length() (translate.lang.common.Common class method), 361
addunit() (translate.storage.properties.stringsfile method), 613	alter_length() (translate.lang.de.de class method), 365
addunit() (translate.storage.properties.stringsutf8file	<pre>alter_length() (translate.lang.el.el class method),</pre>
method), 615 addunit () (translate.storage.properties.xwikifile	366 alter_length() (translate.lang.es.es class method),
method), 616	367
addunit () (translate.storage.properties.XWikiFullPage method), 592	alter_length() (translate.lang.fa.fa class method), 368
addunit () (translate.storage.properties.XWikiPagePropmethod), 594	pe rtie ver_length() (translate.lang.fi.fi class method), 369
addunit () (translate.storage.pypo.pofile method), 622	alter_length() (translate.lang.fr.fr class method), 369
addunit () (translate.storage.qm.qmfile method), 629 addunit () (translate.storage.qph.QphFile method),	alter_length() (translate.lang.gu.gu class method),
634	370
addunit () (translate.storage.rc.rcfile method), 640	alter_length() (translate.lang.he.he class method),
addunit() (translate.storage.subtitles.AdvSubStationAl	
<i>method</i>), 647 addunit () (translate.storage.subtitles.MicroDVDFile	alter_length() (translate.lang.hi.hi class method), 372
method), 648	alter_length() (translate.lang.hy.hy class method),
addunit() (translate.storage.subtitles.SubRipFile	373
method), 650	<pre>alter_length() (translate.lang.ja.ja class method),</pre>

	374		apply_to_strings()	(trans-
alter_	length() (translate.lang.km.km method), 374	class	late.storage.placeables.base.G 515	method),
alter_	length() (translate.lang.kn.kn class	method),	apply_to_strings()	(trans-
	375		late.storage.placeables.base.It	method),
alter_	length() (translate.lang.ko.ko class 376	method),	513 apply_to_strings()	(trans-
alter_	length() (translate.lang.ml.ml class 377	method),	late.storage.placeables.base.Ph 511	method),
alter_	length() (translate.lang.mr.mr	class	apply_to_strings()	(trans-
	method), 377		late.storage.placeables.base.Sub	method),
alter_	length() (translate.lang.ne.ne class	method),	521	
	378		apply_to_strings()	(trans-
alter_	length() (translate.lang.pa.pa class 379	method),	late.storage.placeables.base.X 519	method),
alter_	length() (translate.lang.si.si class	method),	apply_to_strings()	(trans-
alter	381 length() (translate.lang.st.st class	method),	late.storage.placeables.general.Alta method), 523	AttrPlaceable
	381	,	apply_to_strings()	(trans-
alter_	length() (translate.lang.sv.sv class 382	method),	late.storage.placeables.general.XM method), 524	LEntityPlaceable
alter_	length() (translate.lang.ta.ta class	method),	apply_to_strings()	(trans-
	383		late.storage.placeables.general.XM	LTagPlaceable
alter_	length() (translate.lang.te.te class	method),	method), 526	
_	384		<pre>apply_to_strings()</pre>	(trans-
alter_	length() (translate.lang.th.th class 385	method),	late.storage.placeables.interfaces.E method), 528	BasePlaceable
alter_	length() (translate.lang.ug.ug class	method),	apply_to_strings()	(trans-
	385		late.storage.placeables.interfaces.In	nvisiblePlaceable
alter_	length() (translate.lang.ur.ur class	method),	method), 530	
	386	.1 1	<pre>apply_to_strings()</pre>	(trans-
	length() (translate.lang.vi.vi class 387		late.storage.placeables.interfaces.M method), 531	-
alter_	length() (translate.lang.zh.zh class	method),	apply_to_strings()	(trans-
(1	388		late.storage.placeables.interfaces.R	<i>PeplacementPlaceable</i>
	s in translate.lang.am), 358	(*******	method), 533	(tuana
append	file()	(trans-		(trans-
	late.storage.bundleprojstore.BundlePromethod), 414	ojecisiore	late.storage.placeables.interfaces.S method), 535	иоломт шсецые
append	_file()	(trans-	apply_to_strings()	(trans-
	late.storage.projstore.ProjectStore 584	method),	late.storage.placeables.strelem.Stri method), 537	ngElem
apply_	to_strings()	(trans-	apply_to_strings()	(trans-
	late.storage.placeables.base.Bpt 508	method),	late.storage.placeables.terminology method), 539	.TerminologyPlaceable
apply_	to_strings()	(trans-	apply_to_strings()	(trans-
	late.storage.placeables.base.Bx 516	method),	late.storage.placeables.xliff. B pt 541	method),
apply_	to_strings()	(trans-	apply_to_strings()	(trans-
	late.storage.placeables.base.Ept 510	method),	late.storage.placeables.xliff.Bx 546	method),
apply_	to_strings()	(trans-	apply_to_strings()	(trans-
	late.storage.placeables.base.Ex 518	method),	late.storage.placeables.xliff.Ept 543	method),

apply_to_strings() (trans- late.storage.placeables.xliff.Ex method),	blank() (translate.filters.checks.OpenOfficeChecker method), 325
548	blank() (translate.filters.checks.ReducedChecker
apply_to_strings() (trans-	method), 331
late.storage.placeables.xliff.G method), 549	blank() (translate.filters.checks.StandardChecker
apply_to_strings() (trans-	method), 337
late.storage.placeables.xliff.It method), 551	blank() (translate.filters.checks.TermChecker method),
apply_to_strings() (trans-	344
late.storage.placeables.xliff.Ph method),	bn (class in translate.lang.bn), 359
554	Bpt (class in translate.storage.placeables.base), 508
apply_to_strings() (trans-	Bpt (class in translate.storage.placeables.xliff), 541
late.storage.placeables.xliff.Sub method),	brackets()(translate.filters.checks.CCLicenseChecker
552	method), 273
apply_to_strings() (trans-	brackets() (translate.filters.checks.DrupalChecker
late.storage.placeables.xliff.UnknownXML	method), 279
method), 556	brackets() (translate.filters.checks.GnomeChecker
apply_to_strings() (trans-	method), 285
late.storage.placeables.xliff.X method), 544	brackets() (translate.filters.checks.IOSChecker
applytranslation() (in module trans-	method), 291
late.convert.po2dtd), 250	brackets() (translate.filters.checks.KdeChecker method), 297
applytranslation() (in module trans- late.convert.po2prop), 256	brackets() (translate.filters.checks.L20nChecker
ar (class in translate.lang.ar), 358	method), 302
ARBJsonFile (class in translate.storage.jsonl10n), 452	brackets() (translate.filters.checks.LibreOfficeChecker
ARBJsonUnit (class in translate.storage.jsonl10n), 452	method), 308
ArchiveConvertOptionParser (class in trans-	brackets() (translate.filters.checks.MinimalChecker
late.convert.convert), 237	method), 314
attributes (translate.misc.ourdom.Element at-	brackets() (translate.filters.checks.MozillaChecker
tribute), 397	method), 320
D	brackets()(translate.filters.checks.OpenOfficeChecker
В	method), 325
BasePlaceable (class in trans-	$\verb brackets() \textit{(translate.filters.checks.ReducedChecker} $
late.storage.placeables.interfaces), 528	method), 331
batchruntests() (in module trans-	$\verb brackets() \textit{(translate.filters.checks.StandardChecker} $
late.filters.checks), 350	method), 337
blank() (translate.filters.checks.CCLicenseChecker	
method), 273	method), 344
blank() (translate.filters.checks.DrupalChecker	build_checkerconfig() (trans-
method), 279	late.filters.pofilter.FilterOptionParser method), 352
blank() (translate.filters.checks.GnomeChecker	
method), 285	build_idml_store() (in module trans- late.storage.xml_extract.extract), 711
blank() (translate.filters.checks.IOSChecker method),	build_store() (in module trans-
291 blank() (translate.filters.checks.KdeChecker method),	late.storage.xml_extract.extract), 711
297	build_unit_tree() (in module trans-
blank() (translate.filters.checks.L20nChecker	late.storage.xml_extract.unit_tree), 712
method), 302	buildconflictmap() (trans-
blank() (translate.filters.checks.LibreOfficeChecker	late.tools.poconflicts.ConflictOptionParser
method), 308	method), 715
blank() (translate.filters.checks.MinimalChecker	buildfromunit() (translate.storage.base.DictUnit
method), 314	class method), 406
blank() (translate.filters.checks.MozillaChecker	buildfromunit() (trans-
method), 319	late.storage.base.TranslationUnit class
	method), 411

buildfromunit() (trans-	method), 607
late.storage.catkeys.CatkeysUnit class method),	buildfromunit() (trans-
418	late.storage.properties.propunit class method),
buildfromunit() (translate.storage.csvl10n.csvunit	610
class method), 423	buildfromunit() (trans-
buildfromunit() (translate.storage.dtd.dtdunit class method), 429	late.storage.properties.xwikiunit class method), 619
buildfromunit () (translate.storage.html.htmlunit class method), 438	buildfromunit() (translate.storage.pypo.pounit class method), 625
buildfromunit() (translate.storage.ical.icalunit class method), 444	buildfromunit() (translate.storage.qm.qmunit class method), 631
buildfromunit() (translate.storage.ini.iniunit class	buildfromunit() (translate.storage.qph.QphUnit
<pre>method), 449 buildfromunit() (trans-</pre>	<pre>class method), 636 buildfromunit() (translate.storage.rc.rcunit class</pre>
late.storage.jsonl10n.ARBJsonUnit class	method), 642
method), 454	buildfromunit() (trans-
buildfromunit() (trans-	late.storage.subtitles.SubtitleUnit class
late.storage.jsonl10n.GoI18NJsonUnit class	method), 656
method), 459	buildfromunit() (translate.storage.tbx.tbxunit class
buildfromunit() (trans-	method), 661
late.storage.jsonl10n.I18NextUnit class method), 464	buildfromunit() (translate.storage.tiki.TikiUnit class method), 667
buildfromunit() (trans-	<pre>buildfromunit() (translate.storage.tmx.tmxunit</pre>
late.storage.jsonl10n.JsonNestedUnit class	class method), 672
method), 471	buildfromunit() (trans-
<pre>buildfromunit() (trans- late.storage.jsonl10n.JsonUnit class method),</pre>	late.storage.trados.TradosUnit class method), 676
474	buildfromunit() (translate.storage.ts2.tsunit class
buildfromunit() (trans-	method), 683
late.storage.jsonl10n.WebExtensionJsonUnit class method), 479	buildfromunit() (translate.storage.txt.TxtUnit class method), 689
buildfromunit() (translate.storage.lisa.LISAunit	
class method), 484	class method), 694
buildfromunit() (translate.storage.mo.mounit class	
method), 491	late.storage.wordfast.WordfastUnit class
buildfromunit() (trans-	method), 701
late.storage.mozilla_lang.LangUnit class method), 496	buildfromunit () (translate.storage.xliff.xliffunit class method), 708
buildfromunit() (trans-	buildunits() (translate.search.match.matcher
late.storage.omegat.OmegaTUnit class	method), 401
method), 503	buildunits() (trans-
buildfromunit() (trans-	late.search.match.terminologymatcher
late.storage.php.LaravelPHPUnit class	method), 402
method), 560	BundleProjectStore (class in trans-
buildfromunit() (translate.storage.php.phpunit	late.storage.bundleprojstore), 414
class method), 565	Bx (class in translate.storage.placeables.base), 516
buildfromunit() (trans-	Bx (class in translate.storage.placeables.xliff), 546
late.storage.pocommon.pounit class method), 571	С
buildfromunit() (trans-	
late.storage.poxliff.PoXliffUnit class method),	calcstats_old() (in module trans- late.tools.pocount), 718
580	capitalize() (translate.misc.multistring.multistring
buildfromunit() (trans-	method), 389
late.storage.properties.proppluralunit class	capsstart() (translate.lang.af.af class method), 357

capsstart() (translate.lang.am.am class method), 358	character_iter() method), 357	(translate.lang.af.af class
capsstart () (translate.lang.ar.ar class method), 358 capsstart () (translate.lang.bn.bn class method), 359	<pre>character_iter() method), 358</pre>	(translate.lang.am.am class
capsstart() (translate.lang.code_or.code_or class method), 360	<pre>character_iter() method), 359</pre>	(translate.lang.ar.ar class
capsstart() (translate.lang.common.Common class method), 362	<pre>character_iter() method), 359</pre>	(translate.lang.bn.bn class
capsstart () (translate.lang.de.de class method), 365	<pre>character_iter()</pre>	$(translate.lang.code_or.code_or$
capsstart() (translate.lang.el.el class method), 366	class method),	
capsstart() (translate.lang.es.es class method), 367	<pre>character_iter()</pre>	(trans-
capsstart() (translate.lang.fa.fa class method), 368	late.lang.comn	non.Common class method),
capsstart() (translate.lang.fi.fi class method), 369	362	
capsstart() (translate.lang.fr.fr class method), 370	<pre>character_iter()</pre>	(translate.lang.de.de class
capsstart() (translate.lang.gu.gu class method), 370	method), 366	
capsstart() (translate.lang.he.he class method), 371	<pre>character_iter()</pre>	(translate.lang.el.el class
capsstart() (translate.lang.hi.hi class method), 372	method), 366	
capsstart() (translate.lang.hy.hy class method), 373	<pre>character_iter()</pre>	(translate.lang.es.es class
capsstart() (translate.lang.ja.ja class method), 374	method), 367	
capsstart() (translate.lang.km.km class method), 374	<pre>character_iter() method), 368</pre>	(translate.lang.fa.fa class
capsstart() (translate.lang.kn.kn class method), 375	<pre>character_iter()</pre>	(translate.lang.fi.fi class
capsstart() (translate.lang.ko.ko class method), 376	method), 369	
capsstart() (translate.lang.ml.ml class method), 377	<pre>character_iter()</pre>	(translate.lang.fr.fr class
capsstart() (translate.lang.mr.mr class method), 377	method), 370	
capsstart() (translate.lang.ne.ne class method), 378	<pre>character_iter()</pre>	(translate.lang.gu.gu class
capsstart() (translate.lang.pa.pa class method), 379	method), 370	
capsstart() (translate.lang.si.si class method), 381	<pre>character_iter()</pre>	(translate.lang.he.he class
capsstart() (translate.lang.st.st class method), 381	method), 371	
capsstart() (translate.lang.sv.sv class method), 382	<pre>character_iter()</pre>	(translate.lang.hi.hi class
capsstart() (translate.lang.ta.ta class method), 383	method), 372	
capsstart() (translate.lang.te.te class method), 384	<pre>character_iter()</pre>	(translate.lang.hy.hy class
capsstart() (translate.lang.th.th class method), 385	method), 373	
capsstart() (translate.lang.ug.ug class method), 385	<pre>character_iter()</pre>	(translate.lang.ja.ja class
capsstart() (translate.lang.ur.ur class method), 386	method), 374	
capsstart() (translate.lang.vi.vi class method), 387	character_iter()	(translate.lang.km.km class
capsstart () (translate.lang.zh.zh class method), 388	method), 374	
casefold() (translate.misc.multistring.multistring	character_iter()	(translate.lang.kn.kn class
method), 389	method), 375	
categories (translate.filters.checks.TeeChecker at- tribute), 343	character_iter() method), 376	(translate.lang.ko.ko class
categories (translate.filters.checks.UnitChecker at-	<pre>character_iter()</pre>	(translate.lang.ml.ml class
tribute), 350	method), 377	(
CatkeysDialect (class in translate.storage.catkeys),	<pre>character_iter()</pre>	(translate.lang.mr.mr class
415	method), 378	`
CatkeysFile (class in translate.storage.catkeys), 415	character_iter()	(translate.lang.ne.ne class
CatkeysHeader (class in translate.storage.catkeys),	method), 378	` 0
417	character_iter()	(translate.lang.pa.pa class
CatkeysUnit (class in translate.storage.catkeys), 417	method), 379	. 01 1
CCLicenseChecker (class in translate.filters.checks),	<pre>character_iter()</pre>	(translate.lang.si.si class
273	method), 381	, ,
center() (translate.misc.multistring.multistring	character_iter()	(translate.lang.st.st class
method), 389	method), 381	-
	<pre>character_iter()</pre>	(translate.lang.sv.sv class

method), 382			characters() (translate.lang.pa.pa class method),
character_iter()	(translate.lang.ta.ta	class	379
method), 383			characters () (translate.lang.si.si class method), 381
character_iter()	(translate.lang.te.te	class	characters () (translate.lang.st.st class method), 381
method), 384		•	characters() (translate.lang.sv.sv class method),
character_iter()	(translate.lang.th.th	class	382
<i>method</i>), 385	(4	-1	characters () (translate.lang.ta.ta class method), 383
character_iter() <i>method</i>), 385	(translate.lang.ug.ug	class	characters () (translate.lang.te.te class method), 384 characters () (translate.lang.th.th class method), 385
character_iter()	(translate.lang.ur.ur	class	characters() (translate.lang.ug.ug class method),
method), 386	(iransiaic.iang.ur.ur	Ciass	385
character_iter()	(translate.lang.vi.vi	class	characters() (translate.lang.ur.ur class method),
method), 387	` 0		386
character_iter()	(translate.lang.zh.zh	class	characters () (translate.lang.vi.vi class method), 387
method), 388			characters() (translate.lang.zh.zh class method),
characters()(<i>tran</i>	slate.lang.af.af class metho	od), 357	388
	islate.lang.am.am class n	nethod),	check_values() (trans-
358			late.convert.convert.ArchiveConvertOptionParser
	nslate.lang.ar.ar class n	nethod),	method), 237
359		41 1	check_values() (trans-
characters() (<i>trai</i> 359	nslate.lang.bn.bn class n	netnoa),	late.convert.convert.ConvertOptionParser method), 241
	nslate.lang.code_or.code_o	or class	check_values() (trans-
method), 360	isiaie.iang.coae_or.coae_c	n ciass	late.convert.po2moz.MozConvertOptionParser
	slate.lang.common.Comm	on class	method), 253
method), 362	8		check_values() (trans-
* * * * * * * * * * * * * * * * * * * *	nslate.lang.de.de class n	nethod),	late.convert.po2tmx.TmxOptionParser
366			method), 258
	slate.lang.el.el class metho		check_values() (trans-
	nslate.lang.es.es class n	nethod),	late.convert.po2wordfast.WfOptionParser
367		D 260	method), 263
	slate.lang.fa.fa class metho		check_values() (trans-
	slate.lang.fi.fi class method		late.filters.pofilter.FilterOptionParser method), 353
	slate.lang.fr.fr class metho nslate.lang.gu.gu class n		check_values() (trans-
370	isiaie.iaiig.gu.gu ciass ii	ieinou),	late.misc.optrecurse.RecursiveOptionParser
	nslate.lang.he.he class n	nethod).	method), 394
371		, ,	check_values() (trans-
characters()(tran	slate.lang.hi.hi class metho	od), 372	late.tools.poconflicts.ConflictOptionParser
characters() (tra	nslate.lang.hy.hy class n	nethod),	method), 715
373			check_values() (trans-
	slate.lang.ja.ja class metho		late.tools.pogrep.GrepOptionParser method),
	islate.lang.km.km class n	nethod),	719
374		41 1	check_values() (trans-
characters() (<i>tra</i> 375	nslate.lang.kn.kn class n	<i>петпоа)</i> ,	late.tools.porestructure.SplitOptionParser method), 723
	nslate.lang.ko.ko class n	nethod),	check_values() (trans-
376	, ,	, -	late.tools.poterminology.TerminologyOptionParser
	nslate.lang.ml.ml class n	nethod),	method), 726
377	nalata lanc	a a4lc = :N	checker (translate.lang.common.Common attribute),
characters() (<i>trai</i> 378	nslate.lang.mr.mr class n	retnod),	362 checker name (trans-
	nslate.lang.ne.ne class n	nethod)	checker_name (trans- late.filters.checks.CCLicenseChecker at-
378	mana.me.me cuass m	umui,	tribute), 273

•	(trans- attribute),	late.convert.po2wordfast.WfOptionParser method), 263
checker_name late.filters.checks.GnomeChecker a	(trans- attribute),	checkoutputsubdir() (trans- late.filters.pofilter.FilterOptionParser method), 353
285	,,,	checkoutputsubdir() (trans-
checker_name (translate.filters.checks.IOS attribute), 291	SChecker	late.misc.optrecurse.RecursiveOptionParser method), 394
attribute), 297		checkoutputsubdir() (trans- late.tools.poconflicts.ConflictOptionParser
checker_name (translate.filters.checks.L20n	nChecker	method), 715
attribute), 302	(tuana	checkoutputsubdir() (trans-
checker_name late.filters.checks.LibreOfficeChecker	(trans- at-	late.tools.pogrep.GrepOptionParser method), 719
tribute), 308	(trans	checkoutputsubdir() (trans-
*	(trans- ttribute),	late.tools.porestructure.SplitOptionParser method), 723
314	(tuana	checkoutputsubdir() (trans-
·	(trans- ttribute),	late.tools.poterminology.TerminologyOptionParser method), 726
320	(4	cidict (class in translate.misc.dictutils), 388
checker_name late.filters.checks.OpenOfficeChecker	(trans- at-	CJKpunc (translate.lang.common.Common attribute), 361
tribute), 326	(4	classifyunit() (trans-
checker_name	(trans-	late.storage.statistics.Statistics method),
late.filters.checks.ReducedChecker a 331	ttribute),	645 classifyunits() (trans-
checker_name	(trans-	late.storage.statistics.Statistics method),
late.filters.checks.StandardChecker a	`	645
337	,,,	cldr_plural_categories (in module trans-
checker_name	(trans-	late.lang.data), 364
late.filters.checks.StandardUnitChecke tribute), 342	er at-	clean() (translate.tools.poconflicts.ConflictOptionParser method), 715
	nChecker	cleanfile() (in module translate.tools.poclean), 715
attribute), 344		cleanunit () (in module translate.tools.poclean), 715
checker_name	(trans-	cleanup() (translate.storage.bundleprojstore.BundleProjectStore
late.filters.checks.TranslationChecker	at-	method), 414
tribute), 349 checker_name (translate.filters.checks.Uni	tChaokar	clear() (translate.misc.dictutils.cidict method), 389 clear() (translate.storage.oo.unormalizechar
attribute), 350	ichecker	method), 507
CheckerConfig (class in translate.filters.chec	cks), 279	clear_test_dir() (trans-
checkoutputsubdir()	(trans-	late.storage.benchmark.TranslateBenchmarker
late.convert.convert.ArchiveConvertOp	otionParsei	· ·
method), 238		close() (translate.storage.html.htmlfile method), 436
<pre>checkoutputsubdir()</pre>	(trans-	close() (translate.storage.html.POHTMLParser
late.convert.convert.ConvertOptionPar	rser	method), 433
method), 241		close() (translate.storage.project.Project method),
checkoutputsubdir()	(trans-	583
late.convert.po2moz.MozConvertOptio method), 253		code (translate.lang.common.Common attribute), 362 code_or (class in translate.lang.code_or), 360
checkoutputsubdir()	(trans-	combine() (in module translate.convert.accesskey),
late.convert.po2tmx.TmxOptionParser		236
method), 258	(4	Common (class in translate.lang.common), 361
<pre>checkoutputsubdir()</pre>	(trans-	commonpunc (translate.lang.common.Common at-

tribute), 362	late.convert.json2po.json2po method), 248
compendiumconflicts() (trans-	convert_store() (trans-
late.filters.checks.CCLicenseChecker method), 273	late.convert.mozlang2po.lang2po method), 248
compendiumconflicts() (trans-	<pre>convert_store() (translate.convert.php2po.php2po</pre>
late.filters.checks.DrupalChecker method), 279	method), 250 convert_store() (trans-
compendiumconflicts() (trans-	late.convert.po2mozlang.po2lang method),
late.filters.checks.GnomeChecker method),	252
compendiumconflicts() (trans-	convert_store() (translate.convert.po2tiki.po2tiki method), 258
late.filters.checks.IOSChecker method), 291	convert_store() (translate.convert.po2txt.po2txt
compendiumconflicts() (trans-	method), 262
late.filters.checks.KdeChecker method),	convert_store() (translate.convert.rc2po.rc2po method), 269
compendiumconflicts() (trans-	convert_store() (trans-
late.filters.checks.L20nChecker method),	late.convert.resx2po.resx2po method), 269
302 compendiumconflicts() (trans-	convert_store() (translate.convert.tiki2po.tiki2po method), 270
late.filters.checks.LibreOfficeChecker method),	convert_store() (translate.convert.txt2po.txt2po
308	method), 271
compendiumconflicts() (trans-	convert_store() (trans-
late.filters.checks.MinimalChecker method), 314	late.convert.yaml2po.yaml2po method), 272
compendiumconflicts() (trans-	convert_stores() (in module trans-
late.filters.checks.MozillaChecker method), 320	<pre>late.convert.pot2po), 267 convert_unit() (in module trans-</pre>
compendiumconflicts() (trans-	late.convert.sub2po), 269
late.filters.checks.OpenOfficeChecker method), 326	convert_unit() (translate.convert.ical2po.ical2po method), 246
compendiumconflicts() (trans-	<pre>convert_unit() (translate.convert.ini2po.ini2po</pre>
late.filters.checks.ReducedChecker method), 331	<pre>method), 247 convert_unit() (translate.convert.json2po.json2po</pre>
compendiumconflicts() (trans-	method), 248
late.filters.checks.StandardChecker method), 337	<pre>convert_unit() (trans- late.convert.mozlang2po.lang2po method),</pre>
compendiumconflicts() (trans-	249
	<pre>convert_unit() (translate.convert.php2po.php2po method), 250</pre>
compose_mappings() (in module trans-	convert_unit() (trans-
late.storage.xml_extract.misc), 712 con (translate.storage.statsdb.StatsCache attribute), 646	late.convert.po2mozlang.po2lang method), 252
ConflictOptionParser (class in trans-	convert_unit() (translate.convert.po2tiki.po2tiki
late.tools.poconflicts), 715	method), 258
ConsoleColor (class in translate.tools.pocount), 718	convert_unit() (trans-
convert_forward() (trans- late.storage.project.Project method), 583	late.convert.po2yaml.po2yaml method), 267
convert_store() (in module trans- late.convert.sub2po), 269	convert_unit() (translate.convert.rc2po.rc2po method), 269
convert_store() (translate.convert.ical2po.ical2po method), 246	convert_unit() (translate.convert.resx2po.resx2po method), 269
convert_store() (translate.convert.ini2po.ini2po method), 247	convert_unit() (translate.convert.tiki2po.tiki2po method), 270
convert_store() (trans-	convert_unit() (trans-
· · · · · · · · · · · · · ·	trans

- late.convert.yaml2po.yaml2po method), 272 convertcsv() (in module translate.convert.csv2po), convertcsv() (in module translate.convert.csv2tbx), convertcsv() (in module translate.convert.po2csv), 250 convertdtd() (in module translate.convert.dtd2po), 245 convertfile() (translate.convert.csv2tbx.csv2tbx *method*), 245 converthtml() (in module translate.convert.html2po), 246 module converthtml() (in translate.convert.po2html), 251 convert json() module (in translate.convert.json2po), 247 convertmo() (in module translate.tools.pocompile), 715 convertmozillaprop() (in module translate.convert.po2prop), 256 convertmozillaprop() (in module translate.convert.prop2po), 268 convertodf() (in module translate.convert.odf2xliff), 249 convertoo() (in module translate.convert.oo2po), 249 convertoo() (in module translate.convert.oo2xliff), 249 ConvertOptionParser (class translate.convert.convert), 241 convertphp2py() module translate.tools.phppo2pypo), 714 convertpo() (in module translate.convert.po2tmx), convertpo() (in module translate.convert.po2ts), 262 convertpo() module (in translate.convert.po2wordfast), 266 convertpo() (in module translate.convert.po2xliff), convertpo() (in module translate.tools.podebug), 719 convertpo() (in module translate.tools.poswap), 726 convertpot() (in module translate.convert.pot2po), 267 module convertprop() (in translate.convert.prop2po), 268 convertpropunit() (translate.convert.prop2po.prop2po method), 268 convertpy2php() module trans-(in late.tools.pypo2phppo), 730 convertrc() (in module translate.convert.rc2po), 269 convertstore() (translate.convert.csv2po.csv2po method), 245
- convertstore() (translate.convert.prop2po.prop2po method), 268
- convertstrings() (in module translate.convert.po2prop), 257
- convertstrings() (in module translate.convert.prop2po), 268
- convertsub() (in module translate.convert.sub2po), 269
- convertunit() (translate.convert.prop2po.prop2po
 method), 268
- convertxliff() (in module translate.convert.xliff2odf), 271
- convertxliff() (in module translate.convert.xliff2po), 272
- copy () (translate.misc.dictutils.cidict method), 389
- copy () (translate.storage.oo.unormalizechar method), 507
- copy () (translate.storage.placeables.base.Bpt method), 508
- copy () (translate.storage.placeables.base.Bx method), 516
- copy () (translate.storage.placeables.base.Ept method), 510
- copy () (translate.storage.placeables.base.Ex method), 518
- copy () (translate.storage.placeables.base.G method), 515
- copy () (translate.storage.placeables.base.It method), 513
- copy () (translate.storage.placeables.base.Ph method), 511
- copy () (translate.storage.placeables.base.Sub method), 521
- copy () (translate.storage.placeables.base.X method), 519
- copy () (translate.storage.placeables.general.AltAttrPlaceable method), 523
- copy () (translate.storage.placeables.general.XMLEntityPlaceable method), 525
- copy () (translate.storage.placeables.general.XMLTagPlaceable method), 526
- copy () (translate.storage.placeables.interfaces.BasePlaceable method), 528
- copy () (translate.storage.placeables.interfaces.InvisiblePlaceable method), 530
- copy () (translate.storage.placeables.interfaces.MaskingPlaceable method), 531
- copy () (translate.storage.placeables.interfaces.ReplacementPlaceable method), 533
- copy () (translate.storage.placeables.interfaces.SubflowPlaceable method), 535

copy()	(translate.storage.placeables.strelem.StringElem method), 537	<pre>createfilenode() (trans- late.storage.poxliff.PoXliffFile method), 576</pre>
copy()	(translate.storage.placeables.terminology.Terminology.	0 1 00
	method), 539	method), 705
copy()	(translate.storage.placeables.xliff.Bpt method), 541	<pre>creategroup() (translate.storage.poxliff.PoXliffFile</pre>
copy()	(translate.storage.placeables.xliff.Bx method), 546	creategroup() (translate.storage.xliff.xlifffile method), 705
copy()	(translate.storage.placeables.xliff.Ept method), 543	createlanguageNode() (trans- late.storage.lisa.LISAunit method), 484
copy()		createlanguageNode() (trans-
copy()	548 (translate.storage.placeables.xliff.G method),	late.storage.poxliff.PoXliffUnit method), 580
	549	createlanguageNode() (trans-
copy()	(translate.storage.placeables.xliff.It method), 551	<pre>late.storage.qph.QphUnit method), 636 createlanguageNode() (trans-</pre>
copy()		late.storage.tbx.tbxunit method), 661
сору ()	554	createlanguageNode() (trans-
() vgoo	(translate.storage.placeables.xliff.Sub method),	late.storage.tmx.tmxunit method), 672
12 (/	552	createlanguageNode() (trans-
copy()	(translate.storage.placeables.xliff.UnknownXML	late.storage.ts2.tsunit method), 683
	method), 556	createlanguageNode() (trans-
copy()	$(translate.storage.placeables.xliff.X\ method),$	late.storage.xliff.xliffunit method), 708
	544	createParser() (trans-
	(translate.storage.tmx.tmxunit method), 672	late.misc.ourdom.ExpatBuilderNS method),
copyin	put () (in module translate.convert.convert),	397
	244	<pre>createsubfileindex() (trans-</pre>
copyte	mplate() (in module trans-	late.storage.oo.oomultifile method), 507
correc	late.convert.convert), 244 t () (in module translate.filters.autocorrect),	credits() (translate.filters.checks.CCLicenseChecker method), 274
COLICC	273	credits() (translate.filters.checks.DrupalChecker
correc	torigin() (trans-	method), 280
	late.storage.poxliff.PoXliffUnit method), 580	credits() (translate.filters.checks.GnomeChecker method), 285
correc	torigin() (translate.storage.xliff.xliffunit	credits() (translate.filters.checks.IOSChecker
count (method), 708	method), 291 credits() (translate.filters.checks.KdeChecker
count () (translate.misc.multistring.multistring method), 389	method), 297
counta	ccelerators() (in module translate.filters.decoration), 351	credits() (translate.filters.checks.L20nChecker method), 303
countm	atch() (in module translate.filters.helpers), 352	credits() (translate.filters.checks.LibreOfficeChecker method), 308
counts	match() (in module translate.filters.helpers), 352	credits () (translate.filters.checks.MinimalChecker method), 314
countw		credits() (translate.filters.checks.MozillaChecker
	method), 645	method), 320
create	_sample_files() (trans-	credits()(translate.filters.checks.OpenOfficeChecker
	late.storage.benchmark.TranslateBenchmarker	method), 326
	method), 414	credits() (translate.filters.checks.ReducedChecker
create	contextgroup() (trans-	method), 331
	late.storage.poxliff.PoXliffUnit method), 580	credits() (translate.filters.checks.StandardChecker method), 337
create	contextgroup() (trans-late.storage.xliff.xliffunit method), 708	credits() (translate.filters.checks.TermChecker method), 344

csv2po (class in translate.convert.csv2po), 245 csv2tbx (class in translate.convert.csv2tbx), 245 csvfile (class in translate.storage.csv110n), 421	<pre>define_option() (trans- late.tools.poconflicts.ConflictOptionParser method), 715</pre>
csvunit (class in translate.storage.csvl10n), 422 cur (translate.storage.statsdb.StatsCache attribute), 646 cyr21at (in module translate.lang.af), 357	define_option() (trans- late.tools.pogrep.GrepOptionParser method), 719
D	<pre>define_option() (trans- late.tools.porestructure.SplitOptionParser</pre>
de (class in translate.lang.de), 365	method), 723
decode() (translate.storage.pypo.pofile method), 622	define_option() (trans-
decode_header() (in module trans-	late.tools.poterminology.TerminologyOptionParser
late.storage.poparser), 575	method), 726
DefaultDialect (class in translate.storage.csvl10n),	delalttrans() (translate.storage.poxliff.PoXliffUnit
421	method), 580
DefaultDict (translate.storage.base.DictUnit at- tribute), 405	delalttrans() (translate.storage.xliff.xliffunit method), 708
DefaultDict (trans-	delete_range() (trans-
late.storage.jsonl10n.ARBJsonUnit attribute), 454	late.storage.placeables.base.Bpt method), 508
DefaultDict (trans-	delete_range() (trans-
late.storage.jsonl10n.GoI18NJsonUnit at- tribute), 459	late.storage.placeables.base.Bx method), 516
DefaultDict (translate.storage.jsonl10n.118NextUnit	delete_range() (trans-
attribute), 464	late.storage.placeables.base.Ept method),
DefaultDict (trans-	510
late.storage.jsonl10n.JsonNestedUnit at- tribute), 470	delete_range() (trans- late.storage.placeables.base.Ex method),
DefaultDict (translate.storage.jsonl10n.JsonUnit at-	518
tribute), 473	delete_range() (trans-
DefaultDict (trans-	late.storage.placeables.base.G method),
late.storage.jsonl10n.WebExtensionJsonUnit	515 delete range() (trans-
attribute), 478	delete_range() (trans- late.storage.placeables.base.It method),
<pre>define_option() (trans- late.convert.convert.ArchiveConvertOptionParse</pre>	
method), 238	delete_range() (trans-
define_option() (trans-	late.storage.placeables.base.Ph method),
late.convert.convert.ConvertOptionParser	512
method), 241	delete_range() (trans-
define_option() (trans-	$late. storage. placeables. base. Sub \\ method),$
late.convert.po2moz.MozConvertOptionParser	521
method), 253	delete_range() (trans-
<pre>define_option()</pre>	late.storage.placeables.base.X method), 520
method), 258	delete_range() (trans-
define_option() (trans- late.convert.po2wordfast.WfOptionParser	late.storage.placeables.general.AltAttrPlaceable method), 523
method), 263	delete_range() (trans-
<pre>define_option() (trans- late.filters.pofilter.FilterOptionParser method),</pre>	late.storage.placeables.general.XMLEntityPlaceable method), 525
353	delete_range() (trans-
define_option() (trans- late.misc.optrecurse.RecursiveOptionParser	late.storage.placeables.general.XMLTagPlaceable method), 526
method), 394	delete_range() (trans- late.storage.placeables.interfaces.BasePlaceable
	· · · · · · · · · · · · · · · · · ·

method), 528		depth_	_first() (translate.storage.placeal	bles.base.Ex
<pre>delete_range()</pre>	(trans-		method), 518	
	Invisible Place	<i>al</i> depth_	_first() (translate.storage.placed	ıbles.base.G
method), 530	,		method), 515	
<pre>delete_range()</pre>	(trans-		_first() (translate.storage.placed	ables.base.It
late.storage.placeables.interfaces.i	MaskingPlace		method), 513	
method), 532		depth_	_first()	(trans-
<pre>delete_range()</pre>	(trans-		late.storage.placeables.base.Ph	method),
late.storage.placeables.interfaces.	ReplacementP		512	
method), 533		depth_	_first()	(trans-
<pre>delete_range()</pre>	(trans-		late.storage.placeables.base.Sub	method),
late.storage.placeables.interfaces.	Subflow Place a	able	521	
method), 535		depth_	_first() (translate.storage.placed	ables.base.X
<pre>delete_range()</pre>	(trans-		method), 520	
late.storage.placeables.strelem.Str	ingElem	depth_	_first()	(trans-
method), 537			late.storage.placeables.general.Alta	AttrPlaceable
<pre>delete_range()</pre>	(trans-		method), 523	
late.storage.placeables.terminolog	y.Terminology	Placetable	gfirst()	(trans-
method), 539		_	late.storage.placeables.general.XM	LEntityPlaceable
delete_range()	(trans-		method), 525	
late.storage.placeables.xliff.Bpt	method),	depth_	_first()	(trans-
541	,		late.storage.placeables.general.XM	LTagPlaceable
<pre>delete_range()</pre>	(trans-		method), 527	O .
late.storage.placeables.xliff.Bx	method),	depth	_first()	(trans-
546	,,,	1	late.storage.placeables.interfaces.B	,
delete_range()	(trans-		method), 528	
late.storage.placeables.xliff.Ept	method),	depth	_first()	(trans-
543	,,,	1	late.storage.placeables.interfaces.Ii	,
delete_range()	(trans-		method), 530	
late.storage.placeables.xliff.Ex	method),	dept.h	_first()	(trans-
548	,	aop 011_	late.storage.placeables.interfaces.N	,
delete_range() (translate.storage.place	eables.xliff.G		method), 532	zosowa za
method), 549	eao tesimijj. G	dent h	_first()	(trans-
delete_range() (translate.storage.plac	eables xliff It	acpen_	late.storage.placeables.interfaces.R	,
method), 551	eaores.xiijj.11		method), 533	срисстени иссион
delete_range()	(trans-	dent h	_first()	(trans-
late.storage.placeables.xliff.Ph	method),	depen_	late.storage.placeables.interfaces.S	,
554	memou),		method), 535	uojiowi ideedibie
<pre>delete_range()</pre>	(trans-	depth_	_first()	(trans-
late.storage.placeables.xliff.Sub	method),		late.storage.placeables.strelem.Stri	ngElem
553			method), 537	
<pre>delete_range()</pre>	(trans-	depth	_first()	(trans-
late.storage.placeables.xliff.Unkno	wnXML		late.storage.placeables.terminology	.TerminologyPlaceable
method), 556			method), 539	0,
delete_range() (translate.storage.plac	eables.xliff.X	depth	_first()	(trans-
method), 544	33	1 -	late.storage.placeables.xliff.Bpt	method),
depth_first()	(trans-		541	//
late.storage.placeables.base.Bpt	method),	depth	_first() (translate.storage.placed	ıbles.xliff.Bx
509	,,,	1	method), 546	J
depth_first()(translate.storage.placed	ables.base.Bx	dept.h	_first()	(trans-
method), 517			late.storage.placeables.xliff.Ept	method),
depth_first()	(trans-		543	,,
late.storage.placeables.base.Ept	method),	dept.h	_first() (translate.storage.placea	ıbles.xliff.Ex
510	,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,	I	method), 548	······································

depth_	first() (translate.storage.placeables.xliff.G method), 549	dotoat	method), 441 _encoding() (translate.storage.ini.inifile
depth	first() (translate.storage.placeables.xliff.It	uetect.	method), 446
1	method), 551	detect	_encoding() (trans-
depth_	first() (translate.storage.placeables.xliff.Ph		late.storage.jsonl10n.ARBJsonFile method),
	method), 554		452
depth_	first() (trans-	detect.	_encoding() (trans-
	late.storage.placeables.xliff.Sub method), 553		late.storage.jsonl10n.GoI18NJsonFile method), 457
depth		detect	_encoding() (trans-
	late.storage.placeables.xliff.UnknownXML method), 556		late.storage.jsonl10n.118NextFile method), 462
depth_	first() (translate.storage.placeables.xliff.X	detect	_encoding() (trans-
_	method), 545		late.storage.jsonl10n.JsonFile method), 467
destro	y () (translate.convert.convert.ArchiveConvertOp	tion Parser	
	method), 238		late. storage. js on 110 n. Js on Nested File method),
destro	y () (translate.convert.convert.ConvertOptionPar.		469
	method), 241		_encoding() (trans-
	y () (translate.convert.po2moz.MozConvertOption method), 253		method), 477
destro	y () (translate.convert.po2tmx.TmxOptionParser method), 258	detect.	_encoding() (translate.storage.lisa.LISAfile method), 482
destro	y () (translate.convert.po2wordfast.WfOptionPars method), 263	sedretect.	_encoding() (translate.storage.mo.mofile method), 487
destro	y () (translate.filters.pofilter.FilterOptionParser	detect	_encoding() (trans-
	method), 353		late.storage.mozilla_lang.LangStore method),
destro	y () (translate.misc.optrecurse.RecursiveOptionPo		493
	method), 394		_encoding() (trans-
	y () (translate.tools.poconflicts.ConflictOptionPa method), 716		late.storage.omegat.OmegaTFile method), 499
destro	y() (translate.tools.pogrep.GrepOptionParser	detect.	
	method), 719		late.storage.omegat.OmegaTFileTab method),
destro	y () (translate.tools.porestructure.SplitOptionPar		501
doat ro	method), 723 y () (translate.tools.poterminology.TerminologyO		_encoding() (trans- etate.storage.php.LaravelPHPFile method),
destro	y () (transtate:toots:poterminology.1erminology0 method), 727	рионғағы	558 memoa),
	_encoding() (trans- late.storage.base.DictStore method), 404		_encoding() (translate.storage.php.phpfile method), 563
	_encoding() (trans-		_encoding() (trans-
	late.storage.base.TranslationStore method),		late.storage.pocommon.pofile method), 568
	409	detect.	_encoding() (trans-
detect	_encoding() (trans-		late.storage.poxliff.PoXliffFile method), 576
	late.storage.catkeys.CatkeysFile method),	detect.	_encoding() (trans-
	416		late.storage.properties.gwtfile method), 596
detect.	_encoding() (trans-	detect.	_encoding() (trans-
_	late.storage.csvl10n.csvfile method), 421		late.storage.properties.javafile method),
detect.	_encoding() (translate.storage.dtd.dtdfile	1 - 1 - 1	598
do+ = =+	method), 427	aetect.	_encoding() (trans-
	_encoding() (translate.storage.html.htmlfile method), 436		late.storage.properties.javautf16file method), 600
detect.	_encoding() (trans-	detect	_encoding() (trans-
	late.storage.html.POHTMLParser method), 433		late.storage.properties.javautf8file method), 602
detect	_encoding() (translate.storage.ical.icalfile	detect	_encoding() (trans-

	late.storage.properties.joomlafile 603	method),	detect	_encoding(method),687) (translate	e.storage	.txt.TxtFile
	_encoding() late.storage.properties.propfile	(trans- method),		encoding (method), 692		.storage.ı	utx.UtxFile
	605		detect	t_encoding((trans-
	_encoding() late.storage.properties.stringsfile	(trans- method),		late.storage.w 698	ordfast.Wordf	astTMFil	e method),
	613		detect	_encoding() (translate.	storage.	xliff.xlifffile
aetect_	_encoding()	(trans-	al a 4 a a 4	method), 705	(;		*****
	late.storage.properties.stringsutf8file 615	тетоа),	aeteci	t_header() late.storage.c.	*	module	trans-
detect_	_encoding()	(trans-	Diale	ct (class in tran	islate.storage.	ini), 446	
	late.storage.properties.xwikifile	method),	Diale	ct (class in tran	islate.storage.j	propertie	s), 586
	617		Diale	ctDefault (c	lass in transla	te.storage	e.ini), 446
detect_	_encoding()	(trans-	Diale	ctFlex (class	in translate.	storage.p	roperties),
	late.storage.properties.XWikiFullPage method), 592	?	Diale	587 ctGaia (<i>class</i>	in translate.	storage.v	properties).
	_encoding()	(trans-		587			Ι , ,
	late.storage.properties.XWikiPageProp method), 594	`	Diale	ctGwt (class 587	in translate.	storage.p	properties),
	_encoding() (translate.storage.pg method), 622	ypo.pofile		ctInno(<i>class</i> ctJava (<i>class</i>			
detect_	_encoding() (translate.storage.	qm.qmfile		588			
	method), 629		Diale	ctJavaUtf16	(class	in	trans-
detect_	_encoding() (translate.storage.qpl	h.QphFile		late.storage.p	roperties), 588	3	
	method), 634		Diale	ctJavaUtf8	(class	in	trans-
	_encoding() (translate.storag	e.rc.rcfile		late.storage.p	_		
	method), 640		Diale	ctJoomla	(class	in	trans-
	_encoding()	(trans-		late.storage.p			
	late.storage.subtitles.AdvSubStationAl	lphaFile	Diale	ctMozilla	(class	in	trans-
	method), 647			late.storage.p			
detect_	_encoding()	(trans-		cts (in module			
	late.storage.subtitles.MicroDVDFile 649	method),		ctSkype (<i>clas</i> 590	s in translate.	storage.p	properties),
	_encoding() late.storage.subtitles.SubRipFile	(trans-method),	Diale	ctStrings late.storage.p	(class roperties), 590	in)	trans-
	650	,,	Diale	ctStringsUt	_		trans-
detect	_encoding()	(trans-		late.storage.p			
	late.storage.subtitles.SubStationAlpha method), 652			ctXWiki (<i>clas</i> 591			properties),
	_encoding()	(trans-	dialo	gsizes() (tra	ınslate filters c	hecks.L2	0nChecker
	late.storage.subtitles.SubtitleFile	method),		method), 303			
	654	,,	dialo	gsizes()			(trans-
	_encoding() (translate.storage.method), 659	tbx.tbxfile	0.10.10	late.filters.che 320	ecks.MozillaCl	iecker	method),
detect_	_encoding() (translate.storage.tiki method), 665	.TikiStore	dict (translate.storag	ge.catkeys.Cath	keysUnit	attribute),
detect_	_encoding() (translate.storage.tr method), 670	nx.tmxfile	dict (translate.storag 503	e.omegat.Ome	gaTUnit	attribute),
	_encoding()	(trans-	dict(t	ranslate.storage	e.utx.UtxUnit a	attribute)	, 694
	late.storage.trados.TradosTxtTmFile 679	`	dict	(translate.stor tribute), 701			
	_encoding() (translate.storage method), 681	e.ts2.tsfile		tore (class in to		-	

DirDiffer (class in translate.tools.pydiff), 730	280
Directory (class in translate.storage.directory), 426	doublequoting() (trans-
disable_interspersed_args() (trans-	late.filters.checks.GnomeChecker method),
late.convert.convert.ArchiveConvertOptionParser	r 285
method), 238	doublequoting() (trans-
disable_interspersed_args() (trans-	late.filters.checks.IOSChecker method), 291
late.convert.convert.ConvertOptionParser	doublequoting() (trans-
method), 241	late.filters.checks.KdeChecker method),
disable_interspersed_args() (trans-	297
late.convert.po2moz.MozConvertOptionParser	doublequoting() (trans-
method), 253	late.filters.checks.L20nChecker method),
disable_interspersed_args() (trans-	303
late.convert.po2tmx.TmxOptionParser	doublequoting() (trans-
method), 259	$late. filters. checks. Libre Of fice Checker\ method),$
disable_interspersed_args() (trans-	309
late.convert.po2wordfast.WfOptionParser	doublequoting() (trans-
method), 263	late.filters.checks.MinimalChecker method),
disable_interspersed_args() (trans-	314
$late. filters. pofilter. Filter Option Parser\ method),$	doublequoting() (trans-
353	late.filters.checks.MozillaChecker method),
disable_interspersed_args() (trans-	320
late.misc. optrecurse. Recursive Option Parser	doublequoting() (trans-
method), 394	late.filters.checks.OpenOfficeChecker method),
disable_interspersed_args() (trans-	326
late.tools.poconflicts.ConflictOptionParser	doublequoting() (trans-
method), 716	late.filters.checks.ReducedChecker method),
disable_interspersed_args() (trans-	331
late.tools.pogrep.GrepOptionParser method),	doublequoting() (trans-
719	late.filters.checks.StandardChecker method),
disable_interspersed_args() (trans-	337
late. tools. por estructure. Split Option Parser	doublequoting() (trans-
method), 723	late.filters.checks.TermChecker method),
disable_interspersed_args() (trans-	344
late.tools.poterminology.TerminologyOptionParse	
method), 727	late.filters.checks.CCLicenseChecker method),
DiscardUnit, 268	274
distance() (in module translate.search.lshtein), 401	doublespacing() (trans-
do_encoding() (translate.storage.html.htmlfile	late.filters.checks.DrupalChecker method),
method), 436	280
do_encoding() (trans-	doublespacing() (trans-
late.storage.html.POHTMLParser method),	late.filters.checks.GnomeChecker method),
433	286
Document (class in translate.misc.ourdom), 397	doublespacing() (trans-
documentElement (translate.misc.ourdom.Document	late.filters.checks.IOSChecker method), 291
attribute), 397	doublespacing() (trans-
<pre>doreplace() (translate.convert.convert.Replacer</pre>	late.filters.checks.KdeChecker method), 297
DotsProgressBar (class in trans-	doublespacing() (trans-
late.misc.progressbar), 398	late.filters.checks.L20nChecker method),
doublequoting() (trans-	303
late.filters.checks.CCLicenseChecker method),	doublespacing() (trans-
274	$late. filters. checks. Libre Of fice Checker\ method),$
doublequoting() (trans-	309
late.filters.checks.DrupalChecker method),	doublespacing() (trans-

late.filters.checks.MinimalChecker 314	method),	dtdunit (class in translate.storage.dtd), 429	
doublespacing() late.filters.checks.MozillaChecker 320	(trans- method),	el (class in translate.lang.el), 366 elem_at_offset()	(trans-
doublespacing() late.filters.checks.OpenOfficeChecker 326	(trans- method),	late.storage.placeables.base.Bpt 509	method),
doublespacing() late.filters.checks.ReducedChecker	(trans- method),	elem_at_offset() late.storage.placeables.base.Bx 517	(trans- method),
doublespacing() late.filters.checks.StandardChecker	(trans- method),	elem_at_offset()	(trans- method),
337 doublespacing() late.filters.checks.TermChecker	(trans- method),	elem_at_offset()	(trans- method),
344 doublewords() late.filters.checks.CCLicenseChecker	(trans- method),	elem_at_offset()	(trans- method),
doublewords() late.filters.checks.DrupalChecker	(trans- method),	elem_at_offset()	(trans- method),
280 doublewords() late.filters.checks.GnomeChecker	(trans- method),	elem_at_offset()	(trans- method),
286 doublewords() (translate.filters.checks.IC method), 292		elem_at_offset()	(trans- method),
doublewords () (translate.filters.checks.Ka method), 297 doublewords () (translate.filters.checks.L20		elem_at_offset()	(trans- method),
<pre>method), 303 doublewords () late.filters.checks.LibreOfficeChecker</pre>	(trans-	<pre>elem_at_offset()</pre>	(trans- rPlaceable
309 doublewords() late.filters.checks.MinimalChecker	(trans- method),	elem_at_offset()	(trans- EntityPlaceable
315 doublewords() late.filters.checks.MozillaChecker 320	(trans- method),	<pre>elem_at_offset() late.storage.placeables.general.XMLT method), 527 elem_at_offset()</pre>	(trans- TagPlaceable (trans-
doublewords () late.filters.checks.OpenOfficeChecker 326	(trans- method),	late.storage.placeables.interfaces.Bas method), 528 elem_at_offset()	`
doublewords() late.filters.checks.ReducedChecker 332	(trans- method),	late.storage.placeables.interfaces.Invi method), 530 elem_at_offset()	`
doublewords () late.filters.checks.StandardChecker 338	(trans- method),	late.storage.placeables.interfaces.Ma method), 532	skingPlaceable
doublewords () (translate.filters.checks.Ter. method), 345		<pre>elem_at_offset() late.storage.placeables.interfaces.Rep method), 533</pre>	(trans- lacementPlaceabl
DrupalChecker (class in translate.filters.cheddtdfile (class in translate.storage.dtd), 427	ecks), 279	<pre>elem_at_offset()</pre>	(trans- flowPlaceable

method), 535	elem_offset() (translate.storage.placeables.base.X
elem_at_offset() (trans-	method), 520
late. storage. placeables. strelem. String Elem	elem_offset() (trans-
method), 537	late.storage.placeables.general.AltAttrPlaceable
elem_at_offset() (trans-	method), 523
late.storage.placeables.terminology.Terminology	
method), 539	late.storage.placeables.general.XMLEntityPlaceable
elem_at_offset() (trans-	method), 525
late.storage.placeables.xliff.Bpt method),	elem_offset() (trans-
541	late.storage.placeables.general.XMLTagPlaceable
elem_at_offset() (trans-	method), 527
late.storage.placeables.xliff.Bx method),	elem_offset() (trans-
546	late.storage.placeables.interfaces.BasePlaceable
elem_at_offset() (trans-	method), 528
late.storage.placeables.xliff.Ept method),	elem_offset() (trans-
543	late.storage.placeables.interfaces.InvisiblePlaceable
elem_at_offset() (trans-	method), 530
late.storage.placeables.xliff.Ex method),	elem_offset() (trans-
548	late.storage.placeables.interfaces.MaskingPlaceable
elem_at_offset() (trans-	method), 532
late.storage.placeables.xliff.G method), 550	elem_offset() (trans-
elem_at_offset() (trans-	late.storage.placeables.interfaces.ReplacementPlaceable
late.storage.placeables.xliff.It method), 551	method), 533
elem_at_offset() (trans-	elem_offset() (trans-
late.storage.placeables.xliff.Ph method),	late.storage.placeables.interfaces.SubflowPlaceable
554	method), 535
elem_at_offset() (trans-	elem_offset() (trans-
late.storage.placeables.xliff.Sub method),	late.storage.placeables.strelem.StringElem
553	method), 537
elem_at_offset() (trans-	elem_offset() (trans-
late.storage.placeables.xliff.UnknownXML	late.storage.placeables.terminology.TerminologyPlaceable
method), 556	method), 540
elem_at_offset() (trans-	elem_offset() (trans-
late.storage.placeables.xliff.X method), 545	late.storage.placeables.xliff.Bpt method),
elem_offset() (trans-	541
late.storage.placeables.base.Bpt method),	elem_offset() (translate.storage.placeables.xliff.Bx
509	method), 546
elem_offset() (translate.storage.placeables.base.Bx	
method), 517 elem_offset() (trans-	late.storage.placeables.xliff.Ept method), 543
late.storage.placeables.base.Ept method), 510	elem_offset() (translate.storage.placeables.xliff.Ex method), 548
elem_offset() (translate.storage.placeables.base.Ex method), 518	elem_offset() (translate.storage.placeables.xliff.G method), 550
elem_offset() (translate.storage.placeables.base.G	elem_offset() (translate.storage.placeables.xliff.It
method), 515	method), 551
elem_offset() (translate.storage.placeables.base.It	elem_offset() (translate.storage.placeables.xliff.Ph
method), 513	method), 554
elem_offset() (trans-	elem_offset() (trans-
late.storage.placeables.base.Ph method),	late.storage.placeables.xliff.Sub method),
512	553 memoa),
elem_offset() (trans-	elem_offset() (trans-
late.storage.placeables.base.Sub method),	late.storage.placeables.xliff.UnknownXML
521	method), 556
Ç <u>u</u> ı	memous, 550

elem_offset() (translate.storage.placeables.xliff.X method), 394 *method*), 545 enable_interspersed_args() (trans-Element (class in translate.misc.ourdom), 397 late.tools.poconflicts.ConflictOptionParser ElementNotFoundError, 537 method), 716 emails() (translate.filters.checks.CCLicenseChecker enable_interspersed_args() (transmethod), 274 late.tools.pogrep.GrepOptionParser method), (translate.filters.checks.DrupalChecker emails() *method*), 280 enable_interspersed_args() (transemails() (translate.filters.checks.GnomeCheckerlate.tools.porestructure.SplitOptionParser method), 286 method), 723 emails() (translate.filters.checks.IOSChecker enable_interspersed_args() (translate.tools.poterminology.TerminologyOptionParser method), 292 (translate.filters.checks.KdeChecker method), 727 emails() method), 297 (translate.misc.multistring.multistring encode() (translate.filters.checks.L20nChecker emails() method), 390 method), 303 (translate.storage.placeables.base.Bpt encode() emails() (translate.filters.checks.LibreOfficeChecker method), 509 method), 309 (translate.storage.placeables.base.Bx encode() (translate.filters.checks.MinimalChecker method), 517 emails() method), 315 encode() (translate.storage.placeables.base.Ept emails() (translate.filters.checks.MozillaChecker method), 510 method), 320 (translate.storage.placeables.base.Ex encode() emails() (translate.filters.checks.OpenOfficeChecker method), 518 (translate.storage.placeables.base.G *method*), 326 encode() emails() (translate.filters.checks.ReducedChecker method), 515 *method*), 332 encode() (translate.storage.placeables.base.It (translate.filters.checks.StandardCheckermethod), 514 emails() (translate.storage.placeables.base.Ph *method*), 338 encode() (translate.filters.checks.TermChecker method), 512 emails() method), 345 (translate.storage.placeables.base.Sub encode() EMPTY_HTML_ELEMENTS (transmethod), 522 late.storage.html.htmlfile attribute), 435 encode() (translate.storage.placeables.base.X emptyfiletotals() (in module method), 520 translate.storage.statsdb), 646 encode () (translate.storage.placeables.general.AltAttrPlaceable enable interspersed args() method), 523 (trans $late.convert. Archive Convert Option Parser \verb|encode|| () (translate.storage.placeables.general. XMLEntity Placeable) () () (translate.storage.placeables.general. XMLEntity Placeables.general. XMLEntity Placeabl$ *method*), 238 method), 525 enable_interspersed_args() encode() (translate.storage.placeables.general.XMLTagPlaceable (translate.convert.convert.ConvertOptionParser method), 527 encode() (translate.storage.placeables.interfaces.BasePlaceable method), 241enable_interspersed_args() (transmethod), 529 late.convert.po2moz.MozConvertOptionParser encode () (translate.storage.placeables.interfaces.InvisiblePlaceable *method*), 253 *method*), 530 enable_interspersed_args() encode () (translate.storage.placeables.interfaces.MaskingPlaceable (translate.convert.po2tmx.TmxOptionParser method), 532 method), 259encode () (translate.storage.placeables.interfaces.ReplacementPlaceable enable_interspersed_args() (transmethod), 534 late.convert.po2wordfast.WfOptionParser encode() (translate.storage.placeables.interfaces.SubflowPlaceable *method*), 263 method), 535 enable_interspersed_args() (transencode() (translate.storage.placeables.strelem.StringElem late.filters.pofilter.FilterOptionParser method), method), 538 encode () (translate.storage.placeables.terminology.TerminologyPlaceab

762 Index

encode()

method), 540

(translate.storage.placeables.xliff.Bpt

(trans-

enable_interspersed_args()

late.misc.optrecurse.RecursiveOptionParser

	method), 542	endpunc() (translate.filters.checks.KdeChecker
encode	$() \qquad (translate.storage.placeables.xliff.Bx$	method), 297
	method), 546	endpunc() (translate.filters.checks.L20nChecker
encode	0 1	method), 303
	method), 543	endpunc() (translate.filters.checks.LibreOfficeChecker
encode		method), 309
	method), 548	endpunc() (translate.filters.checks.MinimalChecker
encode		method), 315
	method), 550	endpunc() (translate.filters.checks.MozillaChecker
encode	() (translate.storage.placeables.xliff.It method),	method), 321
	551	endpunc() (translate.filters.checks.OpenOfficeChecker
encode	0 1	method), 326
	<i>method</i>), 555	endpunc() (translate.filters.checks.ReducedChecker
encode		method), 332
	method), 553	endpunc() (translate.filters.checks.StandardChecker
encode	() (translate.storage.placeables.xliff.UnknownXM	
	method), 556	endpunc() (translate.filters.checks.TermChecker
encode	() (translate.storage.placeables.xliff.X method),	method), 345
	545	<pre>endswith() (translate.misc.multistring.multistring</pre>
encode	() (translate.storage.properties.Dialect class	method), 390
	method), 586	endwhitespace() (trans-
encode	U 1	late.filters.checks.CCLicenseChecker method),
	class method), 587	274
encode	() (translate.storage.properties.DialectGaia	endwhitespace() (trans-
	class method), 587	late.filters.checks.DrupalChecker method),
encode	() (translate.storage.properties.DialectGwt	280
	class method), 587	endwhitespace() (trans-
encode	() (translate.storage.properties.DialectJava	late.filters.checks.GnomeChecker method),
	class method), 588	286
encode	$(\)\ (translate. storage. properties. Dialect Java Utf 16$	
	class method), 588	late.filters.checks.IOSChecker method), 292
encode	$(\)\ (translate. storage. properties. Dialect Java Utf 8$	endwhitespace() (trans-
	class method), 589	late.filters.checks.KdeChecker method),
encode	() (translate.storage.properties.DialectJoomla	298
	class method), 589	endwhitespace() (trans-
encode	() (translate.storage.properties.DialectMozilla	late.filters.checks.L20nChecker method),
	class method), 590	304
encode	() (translate.storage.properties.DialectSkype	endwhitespace() (trans-
	class method), 590	late.filters.checks.LibreOfficeChecker method),
encode	() (translate.storage.properties.DialectStrings	309
	class method), 590	endwhitespace() (trans-
encode	() (translate.storage.properties.DialectStringsUtf	8 late.filters.checks.MinimalChecker method),
	class method), 591	315
encode	() (translate.storage.properties.DialectXWiki	endwhitespace() (trans-
	class method), 591	late.filters.checks.MozillaChecker method),
encode	() (translate.storage.pypo.pofile method), 622	321
endpun	c () (translate.filters.checks.CCLicenseChecker	endwhitespace() (trans-
	method), 274	late.filters.checks.OpenOfficeChecker method),
endpun		327
_	method), 280	endwhitespace() (trans-
endpun		late.filters.checks.ReducedChecker method),
-	method), 286	332
endpun		endwhitespace() (trans-
	method), 292	late.filters.checks.StandardChecker method),

338	method), 298
endwhitespace() (trans-	escapes() (translate.filters.checks.L20nChecker
late.filters.checks.TermChecker method),	method), 304
345	$\verb escapes () \textit{(translate.filters.checks.LibreOfficeChecker} $
<pre>entitydecode() (in module translate.misc.quote),</pre>	method), 310
399	escapes() (translate.filters.checks.MinimalChecker
<pre>entityencode() (in module translate.misc.quote),</pre>	method), 315
399	escapes() (translate.filters.checks.MozillaChecker
Ept (class in translate.storage.placeables.base), 510	method), 321
Ept (class in translate.storage.placeables.xliff), 543	escapes() (translate.filters.checks.OpenOfficeChecker
$\verb error () (translate.convert.convert.Archive Convert Option \\$	
method), 238	escapes() (translate.filters.checks.ReducedChecker
$\verb error () (translate.convert.convert.ConvertOptionParser $	method), 332
method), 241	escapes() (translate.filters.checks.StandardChecker
$\verb error () (translate.convert.po2moz.MozConvertOptionPartition Particle ()) (translate.convert.po2moz.MozConvertOptionPartition Particle ()) (translate.convert.po2moz.MozConvertOptionPartition Particle ()) (translate.convert.po2moz.MozConvertOptionPartition Partition Partition$	
method), 253	escapes() (translate.filters.checks.TermChecker
error() (translate.convert.po2tmx.TmxOptionParser	method), 345
method), 259	ethiopicpunc (translate.lang.common.Common at-
$\verb error () \textit{(translate.convert.po2wordfast.WfOptionParser} $	
method), 263	Ex (class in translate.storage.placeables.base), 518
error() (translate.filters.pofilter.FilterOptionParser	Ex (class in translate.storage.placeables.xliff), 547
method), 353	<pre>expandtabs() (translate.misc.multistring.multistring</pre>
$\verb error () (translate.misc.optrecurse.RecursiveOptionParse) $	
method), 394	expansion_factors (in module translate.lang.data),
$\verb error () \textit{(translate.tools.poconflicts.ConflictOptionParse} \\$	
method), 716	ExpatBuilderNS (class in translate.misc.ourdom),
error() (translate.tools.pogrep.GrepOptionParser	397
method), 720	<pre>export_file() (translate.storage.project.Project</pre>
$\verb error () (translate.tools.porestructure.SplitOptionParser $	
method), 723	<pre>extendtm() (translate.search.match.matcher method),</pre>
error() (translate.tools.poterminology.TerminologyOptic	
method), 727	extendtm() (translate.search.match.terminologymatche
es (class in translate.lang.es), 367	method), 402
escape() (in module translate.storage.trados), 675	Extensions (translate.storage.base.TranslationStore
escape_help_text() (in module trans-	attribute), 408
late.storage.oo), 505	extract() (in module translate.convert.accesskey),
escape_text() (in module translate.storage.oo), 505	237
	extract() (in module translate.misc.quote), 399
late.storage.rc), 639	extract_msgid_comment() (in module trans-
escape_to_rc() (in module translate.storage.rc),	late.storage.pocommon), 568
639	extractwithoutquotes() (in module trans-
escapecontrols() (in module translate.misc.quote),	late.misc.quote), 399
399	F
escapeforpo() (in module translate.storage.pypo),	
621	fa (class in translate.lang.fa), 368
escapes () (translate.filters.checks.CCLicenseChecker	fallback_detection() (trans-
method), 275	late.storage.base.DictStore method), 404
escapes() (translate.filters.checks.DrupalChecker	fallback_detection() (trans-
method), 281	late.storage.base.TranslationStore method),
escapes() (translate.filters.checks.GnomeChecker	409
method), 286	fallback_detection() (trans-
escapes() (translate.filters.checks.IOSChecker	late.storage.catkeys.CatkeysFile method),
method), 292	416
escapes() (translate.filters.checks.KdeChecker	

<pre>fallback_detection() late.storage.csvl10n.csvfile method),</pre>	421	fallback_detection() late.storage.properties.javafile	(trans- method),
fallback_detection()	(trans-	598	(4
late.storage.dtd.dtdfile method), 427	(tuana	fallback_detection()	(trans-
fallback_detection()	(trans-	late.storage.properties.javautf16file	method),
late.storage.html.htmlfile method), 43		600	(tuana
fallback_detection()	(trans-	fallback_detection()	(trans-
late.storage.html.POHTMLParser 433	method),	late.storage.properties.javautf8file 602	method),
<pre>fallback_detection()</pre>		fallback_detection()	(trans-
late.storage.ical.icalfile method), 441		late.storage.properties.joomlafile	method),
<pre>fallback_detection()</pre>	(trans-	603	,
late.storage.ini.inifile method), 447		fallback_detection()	(trans-
<pre>fallback_detection()</pre>	(trans-	late.storage.properties.propfile	method),
late.storage.jsonl 10 n. ARB Json File	method),	605	
452		<pre>fallback_detection()</pre>	(trans-
<pre>fallback_detection() late.storage.jsonl10n.GoI18NJsonFile</pre>	e (trans-	late.storage.properties.stringsfile 613	method),
method), 457		fallback_detection()	(trans-
<pre>fallback_detection()</pre>	(trans-	late.storage.properties.stringsutf8file	method),
late.storage.jsonl10n.I18NextFile	method),	615	
462		fallback_detection()	(trans-
<pre>fallback_detection()</pre>	(trans-	late.storage.properties.xwikifile	method),
late.storage.jsonl10n.JsonFile method	d), 467	617	
fallback_detection()	(trans-	fallback_detection()	(trans-
late.storage.jsonl10n.JsonNestedFile 469		late.storage.properties.XWikiFullPage method), 592	!
<pre>fallback_detection()</pre>	(trans-	fallback_detection()	(trans-
late.storage.jsonl10n.WebExtensionJs method), 477	`	late.storage.properties.XWikiPagePromethod), 594	`
fallback_detection()	(trans-	fallback_detection()	(trans-
late.storage.lisa.LISAfile method), 48	2	late.storage.pypo.pofile method), 622	`
fallback_detection()	(trans-	fallback_detection()	(trans-
late.storage.mo.mofile method), 488	(*	late.storage.qm.qmfile method), 629	(
fallback_detection()	(trans-	fallback_detection()	(trans-
late.storage.mozilla_lang.LangStore	method),	late.storage.qph.QphFile method), 63-	,
494	,,	fallback_detection() (translate.storage	
fallback_detection()	(trans-	method), 640	J
late.storage.omegat.OmegaTFile	method),	fallback_detection()	(trans-
499	,,,	late.storage.subtitles.AdvSubStationA	`
<pre>fallback_detection()</pre>	(trans-	method), 647	<i>p</i> · · · · · ·
late.storage.omegat.OmegaTFileTab	method),	fallback_detection()	(trans-
501	memou),	late.storage.subtitles.MicroDVDFile	method),
fallback_detection()	(trans-	649	memou),
late.storage.php.LaravelPHPFile	method),	fallback_detection()	(trans-
558	memou),	late.storage.subtitles.SubRipFile	method),
fallback_detection()	(trans-	650	memou),
late.storage.php.phpfile method), 563		fallback_detection()	(trans-
fallback_detection()	(trans-	late.storage.subtitles.SubStationAlpha	
late.storage.pocommon.pofile method		method), 652	1116
		fallback_detection()	(trans-
<pre>fallback_detection() late.storage.poxliff.PoXliffFile method</pre>	(trans-	late.storage.subtitles.SubtitleFile	method),
fallback_detection()	(trans-	654	memou),
late.storage.properties.gwtfile method	`	fallback_detection()	(trans-
idie.siorage.properties.gwijne memot	ı,, 270	Tallnacy acception ()	(iiuiis-

late.storage.tbx.tbxfile method), 659	late.filters.checks.MinimalChecker method),
<pre>fallback_detection() (trans-</pre>	315
late.storage.tiki.TikiStore method), 665	filepaths() (translate.filters.checks.MozillaChecker
fallback_detection() (trans-	method), 321
late.storage.tmx.tmxfile method), 670	filepaths() (trans-
fallback_detection() (trans-	late.filters.checks.OpenOfficeChecker method),
late.storage.trados.TradosTxtTmFile method),	327
679	filepaths() (trans-
<pre>fallback_detection() (translate.storage.ts2.tsfile</pre>	late.filters.checks.ReducedChecker method), 333
fallback_detection() (trans-	filepaths() (trans-
late.storage.txt.TxtFile method), 687	late.filters.checks.StandardChecker method),
fallback_detection() (trans-	338
late.storage.utx.UtxFile method), 692	filepaths() (translate.filters.checks.TermChecker
fallback_detection() (trans-	method), 345
late.storage.wordfast.WordfastTMFile method),	filestatestats() (trans-
698	late.storage.statsdb.StatsCache method),
fallback_detection() (trans-	646
late.storage.xliff.xlifffile method), 705 feed() (translate.storage.html.htmlfile method), 436	filestats() (translate.storage.statsdb.StatsCache method), 646
feed() (translate.storage.html.POHTMLParser method), 433	filetotals() (translate.storage.statsdb.StatsCache method), 646
fi (class in translate.lang.fi), 369	fill() (translate.storage.pypo.PoWrapper method),
FIELDNAMES (in module translate.storage.catkeys), 420	621
FIELDNAMES_HEADER (in module trans-	filteraccelerators() (in module trans-
late.storage.catkeys), 420	late.filters.prefilters), 355
FIELDNAMES_HEADER_DEFAULTS (in module trans-	filteraccelerators_by_list() (trans-
late.storage.catkeys), 420	late.filters.checks.CCLicenseChecker method),
<pre>file_iter() (translate.storage.directory.Directory</pre>	275
method), 426	filteraccelerators_by_list() (trans-
file_iter() (translate.storage.zip.ZIPFile method), 714	late.filters.checks.DrupalChecker method), 281
filechecks() (translate.storage.statsdb.StatsCache	filteraccelerators_by_list() (trans-
method), 646	late.filters.checks.GnomeChecker method),
FileDiffer (class in translate.tools.pydiff), 730	287
FileExistsInProjectError,584	filteraccelerators_by_list() (trans-
FileNotInProjectError,584	late.filters.checks.IOSChecker method), 292
	filteraccelerators_by_list() (trans-
late.filters.checks.CCLicenseChecker method), 275	late.filters.checks.KdeChecker method), 298
filepaths() (translate.filters.checks.DrupalChecker	filteraccelerators_by_list() (trans-
method), 281	late.filters.checks.L20nChecker method),
filepaths() (translate.filters.checks.GnomeChecker	304
method), 287	filteraccelerators_by_list() (trans-
filepaths() (translate.filters.checks.IOSChecker method), 292	late.filters.checks.LibreOfficeChecker method), 310
filepaths() (translate.filters.checks.KdeChecker	filteraccelerators_by_list() (trans-
method), 298	late.filters.checks.MinimalChecker method),
filepaths() (translate.filters.checks.L20nChecker	315
method), 304	filteraccelerators_by_list() (trans-
filepaths() (trans-	late.filters.checks.MozillaChecker method),
late.filters.checks.LibreOfficeChecker method),	321
310	filteraccelerators_by_list() (trans-
filepaths() (trans-	late.filters.checks.OpenOfficeChecker method),

327		filtertestmethod() (in module trans-
filteraccelerators_by_list()	(trans-	late.filters.helpers), 352
late.filters.checks.ReducedChecker 333	method),	filtervariables() (in module trans- late.filters.prefilters), 356
<pre>filteraccelerators_by_list()</pre>	(trans-	filterwordswithpunctuation() (in module
late.filters.checks.StandardChecker 338	method),	<pre>translate.filters.prefilters), 356 finalizetempoutputfile() (trans-</pre>
filteraccelerators_by_list()	(trans-	late.convert.convert.ArchiveConvertOptionParser
late.filters.checks.StandardUnitCheck method), 342	ker	<pre>method), 238 finalizetempoutputfile() (trans-</pre>
filteraccelerators_by_list()	(trans-	late.convert.convert.ConvertOptionParser
late. filters. checks. Term Checker	method),	method), 242
345		finalizetempoutputfile() (trans-
$filter accelerators_by_list() \\ \textit{late.filters.checks.TranslationChecke}$	(trans- r method),	late.convert.po2moz.MozConvertOptionParser method), 254
349		finalizetempoutputfile() (trans-
<pre>filteraccelerators_by_list() late.filters.checks.UnitChecker</pre>	(trans- method),	late.convert.po2tmx.TmxOptionParser method), 259
350		finalizetempoutputfile() (trans-
filtercount() (in module translate.filter 352	rs.helpers),	late.convert.po2wordfast.WfOptionParser method), 264
FilterFailure, 285		finalizetempoutputfile() (trans-
<pre>filterinputformats()</pre>	(trans-	$late. filters. pofilter. Filter Option Parser\ method),$
late.convert.convert.ArchiveConvert(method), 238	OptionParse	r 353 finalizetempoutputfile() (trans-
<pre>filterinputformats() late.convert.convert.ConvertOptionPage</pre>	(trans- arser	late.misc.optrecurse.RecursiveOptionParser method), 394
method), 242		finalizetempoutputfile() (trans-
<pre>filterinputformats() late.convert.po2moz.MozConvertOpt</pre>	(trans- ionParser	late.tools.poconflicts.ConflictOptionParser method), 716
method), 253		finalizetempoutputfile() (trans-
<pre>filterinputformats()</pre>	(trans-	late. tools. pogrep. Grep Option Parser method),
late.convert.po2tmx.TmxOptionParse method), 259	er -	720 finalizetempoutputfile() (trans-
filterinputformats() late.convert.po2wordfast.WfOptionPo	(trans-	late.tools.porestructure.SplitOptionParser method), 723
method), 263		finalizetempoutputfile() (trans-
FilterOptionParser (class in late.filters.pofilter), 352	trans-	
filteroutputoptions()	(trans-	find() (translate.misc.multistring.multistring method),
late.convert.convert.ArchiveConvert(r 390
method), 238	(tuans	find() (translate.storage.placeables.base.Bpt method), 509
filteroutputoptions() late.convert.convert.ConvertOptionPo	(trans- arser	find() (translate.storage.placeables.base.Bx method),
<pre>method), 242 filteroutputoptions()</pre>	(trans-	517 find() (translate.storage.placeables.base.Ept method),
late.convert.po2moz.MozConvertOpt	`	510
method), 253		find() (translate.storage.placeables.base.Ex method),
filteroutputoptions() late.convert.po2tmx.TmxOptionParse	(trans-	518 find() (translate.storage.placeables.base.G method),
method), 259	(40	515
filteroutputoptions() late.convert.po2wordfast.WfOptionPo	(trans- arser	find() (translate.storage.placeables.base.It method), 514
method), 263		find() (translate.storage.placeables.base.Ph method),

	512		method), 587	
find()	(translate.storage.placeables.base.Sub m	ethod),	<pre>find_delimiter()</pre>	(trans-
	522		late.storage.properties.DialectGwt	class
find()	(translate.storage.placeables.base.X m	ethod),	method), 587	,
	520		<pre>find_delimiter()</pre>	(trans-
find()	(translate.storage.placeables.general.Alta method), 523	AttrPlace	eable late.storage.properties.DialectJava method), 588	class
find()	(translate.storage.placeables.general.XM	LEntityF	<i>Placeab<u>l</u>e</i> delimiter()	(trans-
	method), 525		late.storage.properties.Dialect Java U	tf16
find()	(translate. storage. placeables. general. XM	LTagPla	ceable class method), 588	
	method), 527		<pre>find_delimiter()</pre>	(trans-
find()	(translate.storage.placeables.interfaces.E method), 529	BasePlac	eable late.storage.properties.DialectJavaU method), 589	tf8 class
find()	(translate.storage.placeables.interfaces.I	nvisibleF		(trans-
	method), 530		late.storage.properties.DialectJooml	
find()	(translate.storage.placeables.interfaces.M	1asking F		
	method), 532		<pre>find_delimiter()</pre>	(trans-
find()	(translate.storage.placeables.interfaces.R	?eplacem	nentPlaceal <mark>lkt</mark> e.storage.properties.DialectMozill	a class
	method), 534		method), 590	
find()	(translate.storage.placeables.interfaces.S	ubflowP	<i>lācenbl<u>e</u></i> delimiter()	(trans-
	method), 535		late. storage. properties. Dialect Skype	class
find()	(translate. storage. placeables. strelem. Stri	ngElem	method), 590	
	method), 538		<pre>find_delimiter()</pre>	(trans-
find()	(translate.storage.placeables.terminology method), 540	.Termino	ologyPlace lutte .storage.properties.DialectString. method), 590	s class
find()	(translate.storage.placeables.xliff.Bpt ma	ethod),	find_delimiter()	(trans-
	542		late.storage.properties.DialectString.	sUtf8
find()	(translate.storage.placeables.xliff.Bx m	ethod),	class method), 591	
	547		<pre>find_delimiter()</pre>	(trans-
find()	(translate.storage.placeables.xliff.Ept me 543	ethod),	late.storage.properties.DialectXWiki method), 591	class
find()	(translate.storage.placeables.xliff.Ex ma	ethod).	find_dom_root() (in module	trans-
()	548	,,	late.storage.xml_extract.generate), 7	
find()	(translate.storage.placeables.xliff.G ma	ethod).	find_elems_with()	(trans-
- ()	550	,,,	late.storage.placeables.base.Bpt	method),
find()	(translate.storage.placeables.xliff.It ma	ethod),	509	,,
	551		find_elems_with()	(trans-
find()	(translate.storage.placeables.xliff.Ph ma	ethod),	late.storage.placeables.base.Bx	method),
	555		517	
find()	(translate.storage.placeables.xliff.Sub ma	ethod),	find_elems_with()	(trans-
	553		late.storage.placeables.base.Ept	method),
find()	(translate.storage.placeables.xliff.Unknow	vnXML	510	
	method), 556		find_elems_with()	(trans-
find()	(translate.storage.placeables.xliff.X m	ethod),	late.storage.placeables.base.Ex	method),
	545		518	
find_a	all () (in module translate.misc.quote), 3	99	find_elems_with()	(trans-
find_c	delimiter()	(trans-	late.storage.placeables.base.G	method),
	late.storage.properties.Dialect class me	ethod),	515	
	586		find_elems_with()	(trans-
find_c	delimiter()	(trans-	late.storage.placeables.base.It	method),
	late.storage.properties.DialectFlex	class	514	
	method), 587		find_elems_with()	(trans-
find_c	delimiter()	(trans-	late.storage.placeables.base.Ph	method),
	late.storage.properties.DialectGaia	class	512	

```
555
find elems with()
                                              (trans-
         late.storage.placeables.base.Sub
                                            method),
                                                      find_elems_with()
                                                                                                     (trans-
                                                                late.storage.placeables.xliff.Sub
                                                                                                   method),
                                                                553
find_elems_with()
                                              (trans-
         late.storage.placeables.base.X
                                            method),
                                                       find elems with()
                                                                                                     (trans-
                                                                late.storage.placeables.xliff.UnknownXML
                                              (trans-
                                                                method), 556
find elems with()
         late.storage.placeables.general.AltAttrPlaceable find elems with()
                                                                                                     (trans-
         method), 523
                                                                late.storage.placeables.xliff.X method), 545
find_elems_with()
                                              (trans- find_matches() (in module translate.tools.pogrep),
         late.storage.placeables.general.XMLEntityPlaceable
                                                                722
                                                       find_placeable_dom_tree_roots() (in mod-
         method), 525
                                                                      translate.storage.xml extract.generate),
find_elems_with()
                                              (trans-
                                                                ule
         late.storage.placeables.general.XMLTagPlaceable
                                                                711
         method), 527
                                                       findaccelerators()
                                                                                    (in
                                                                                           module
                                                                                                      trans-
find_elems_with()
                                              (trans-
                                                                late.filters.decoration), 351
         late.storage.placeables.interfaces.BasePlaceable findid() (translate.storage.base.DictStore method),
         method), 529
                                                                404
find_elems_with()
                                              (trans- findid()
                                                                      (translate.storage.base.TranslationStore
         late.storage.placeables.interfaces.InvisiblePlaceable
                                                                method), 409
        method), 530
                                                       findid()
                                                                        (translate.storage.catkeys.CatkeysFile
find elems with()
                                              (trans-
                                                                method), 416
         late.storage.placeables.interfaces.MaskingPlaceablendid() (translate.storage.csvl10n.csvfile method),
         method), 532
                                                                421
find_elems_with()
                                                      findid() (translate.storage.dtd.dtdfile method), 427
                                              (trans-
         late.storage.placeables.interfaces.ReplacementPlaceabled() (translate.storage.html.htmlfile method), 436
         method), 534
                                                       findid()
                                                                      (translate.storage.html.POHTMLParser
                                              (trans-
                                                                method), 433
find_elems_with()
         late.storage.placeables.interfaces.SubflowPlaceableindid() (translate.storage.ical.icalfile method), 442
                                                       findid() (translate.storage.ini.inifile method), 447
         method), 535
find_elems_with()
                                              (trans-
                                                      findid()
                                                                      (translate.storage.jsonl10n.ARBJsonFile
         late.storage.placeables.strelem.StringElem
                                                                method), 452
                                                       findid() (translate.storage.jsonl10n.GoI18NJsonFile
         method), 538
find_elems_with()
                                                                method), 457
                                              (trans-
         late.storage.placeables.terminology.TerminologyPfaceable()
                                                                       (translate.storage.jsonl10n.I18NextFile
        method), 540
                                                                method), 462
find elems with()
                                              (trans-
                                                       findid()
                                                                          (translate.storage.jsonl10n.JsonFile
         late.storage.placeables.xliff.Bpt
                                            method),
                                                                method), 467
         542
                                                       findid()
                                                                    (translate.storage.jsonl10n.JsonNestedFile
find_elems_with()
                                                                method), 469
                                              (trans-
         late.storage.placeables.xliff.Bx
                                                       findid()(translate.storage.jsonl10n.WebExtensionJsonFile
                                            method),
         547
                                                                method), 477
                                                       findid() (translate.storage.lisa.LISAfile method), 482
find elems with()
                                              (trans-
         late.storage.placeables.xliff.Ept
                                                       findid() (translate.storage.mo.mofile method), 488
                                            method),
                                                                    (translate.storage.mozilla_lang.LangStore
                                                       findid()
find_elems_with()
                                              (trans-
                                                                method), 494
                                                                        (translate.storage.omegat.OmegaTFile
         late.storage.placeables.xliff.Ex
                                            method),
                                                      findid()
         548
                                                                method), 499
find_elems_with()
                                              (trans-
                                                       findid()
                                                                    (translate.storage.omegat.OmegaTFileTab
         late.storage.placeables.xliff.G method), 550
                                                                method), 501
                                                                       (translate.storage.php.LaravelPHPFile
find_elems_with()
                                              (trans-
                                                       findid()
         late.storage.placeables.xliff.It method), 551
                                                                method), 558
find_elems_with()
                                              (trans-
                                                       findid() (translate.storage.php.phpfile method), 563
         late.storage.placeables.xliff.Ph
                                            method), findid()(translate.storage.pocommon.pofile method),
```

560	
568	findunit() (translate.storage.base.TranslationStore
findid() (translate.storage.poxliff.PoXliffFile method), 576	method), 409 findunit() (translate.storage.catkeys.CatkeysFile
findid() (translate.storage.properties.gwtfile	findunit() (translate.storage.catkeys.CatkeysFile method), 416
method), 596	findunit() (translate.storage.csvl10n.csvfile
findid() (translate.storage.properties.javafile	method), 421
method), 598	findunit () (translate.storage.dtd.dtdfile method), 427
findid() (translate.storage.properties.javautf16file	findunit() (translate.storage.html.htmlfile method),
method), 600	436
findid() (translate.storage.properties.javautf8file	findunit () (translate.storage.html.POHTMLParser
method), 602	method), 433
findid() (translate.storage.properties.joomlafile	findunit() (translate.storage.ical.icalfile method),
method), 603	442
findid() (translate.storage.properties.propfile	findunit () (translate.storage.ini.inifile method), 447
method), 605	findunit() (translate.storage.jsonl10n.ARBJsonFile
findid() (translate.storage.properties.stringsfile	method), 452
method), 613	findunit() (translate.storage.jsonl10n.GoI18NJsonFile
findid() (translate.storage.properties.stringsutf8file	method), 457
method), 615	findunit() (translate.storage.jsonl10n.118NextFile
findid() (translate.storage.properties.xwikifile	method), 462
method), 617	findunit() (translate.storage.jsonl10n.JsonFile
findid() (translate.storage.properties.XWikiFullPage	method), 467
method), 592	findunit()(translate.storage.jsonl10n.JsonNestedFile
findid()(translate.storage.properties.XWikiPagePrope	rties method), 469
method), 594	findunit()(translate.storage.jsonl10n.WebExtensionJsonFile
findid() (translate.storage.pypo.pofile method), 622	method), 477
findid() (translate.storage.qm.qmfile method), 629	findunit() (translate.storage.lisa.LISAfile method),
findid() (translate.storage.qph.QphFile method), 634	482
findid() (translate.storage.rc.rcfile method), 640	findunit() (translate.storage.mo.mofile method), 488
findid()(translate.storage.subtitles.AdvSubStationAlph	na Fil nadunit () (translate.storage.mozilla_lang.LangStore
method), 647	method), 494
<pre>findid() (translate.storage.subtitles.MicroDVDFile</pre>	findunit() (translate.storage.omegat.OmegaTFile
method), 649	method), 499
findid() (translate.storage.subtitles.SubRipFile	findunit()(translate.storage.omegat.OmegaTFileTab
method), 650	method), 501
findid() (translate.storage.subtitles.SubStationAlphaFi	
method), 652	method), 558
findid() (translate.storage.subtitles.SubtitleFile	findunit() (translate.storage.php.phpfile method),
method), 654	563
findid() (translate.storage.tbx.tbxfile method), 659	findunit() (translate.storage.pocommon.pofile
findid() (translate.storage.tiki.TikiStore method), 665	method), 568
findid() (translate.storage.tmx.tmxfile method), 670	findunit() (translate.storage.poxliff.PoXliffFile
findid() (translate.storage.trados.TradosTxtTmFile	method), 576
method), 679	findunit() (translate.storage.properties.gwtfile
findid() (translate.storage.ts2.tsfile method), 681	method), 596
findid() (translate.storage.txt.TxtFile method), 687	findunit() (translate.storage.properties.javafile
findid() (translate.storage.utx.UtxFile method), 692	method), 598
findid() (translate.storage.wordfast.WordfastTMFile	findunit () (translate.storage.properties.javautf16file
method), 698	method), 600
findid() (translate.storage.xliff.xlifffile method), 705	findunit () (translate.storage.properties.javautf8file
findmarkedvariables() (in module trans-	method), 602
late.filters.decoration), 351	findunit() (translate.storage.properties.joomlafile
findunit() (translate.storage.base.DictStore	method), 603
method), 404	findunit() (translate.storage.properties.propfile

method), 605	436
findunit() (translate.storage.properties.stringsfile method), 613	findunits() (translate.storage.html.POHTMLParser method), 434
findunit () (translate.storage.properties.stringsutf8file method), 615	findunits() (translate.storage.ical.icalfile method), 442
findunit() (translate.storage.properties.xwikifile method), 617	findunits() (translate.storage.ini.inifile method), 447
${\tt findunit()} \ ({\it translate.storage.properties.} XWikiFull Page$	findunits() (trans-
method), 592	$late. storage. js on l10 n. ARB Js on File \qquad method),$
findunit () (translate.storage.properties.XWikiPageProperties.XWik	perties 452 findunits() (trans-
findunit() (translate.storage.pypo.pofile method), 622	late.storage.jsonl10n.GoI18NJsonFile method), 457
findunit() (translate.storage.qm.qmfile method), 629	findunits() (translate.storage.jsonl10n.I18NextFile
findunit() (translate.storage.qph.QphFile method),	method), 462
634	findunits() (translate.storage.jsonl10n.JsonFile
findunit() (translate.storage.rc.rcfile method), 640	method), 467
${\tt findunit()} \ ({\it translate.storage.subtitles.} Adv Sub Station Adv Sub S$	
method), 647	$late. storage. js on 110 n. Js on Nested File \ method),$
findunit()(translate.storage.subtitles.MicroDVDFile	469
method), 649	findunits() (trans-
findunit() (translate.storage.subtitles.SubRipFile method), 651	late.storage.jsonl10n.WebExtensionJsonFile method), 477
<pre>findunit() (translate.storage.subtitles.SubStationAlpha</pre>	Hilmodunits () (translate.storage.lisa.LISAfile method), 482
findunit() (translate.storage.subtitles.SubtitleFile method), 654	findunits() (translate.storage.mo.mofile method), 488
findunit() (translate.storage.tbx.tbxfile method), 659	findunits() (trans-
findunit() (translate.storage.tiki.TikiStore method), 665	late.storage.mozilla_lang.LangStore method), 494
665 findunit() (translate.storage.tmx.tmxfile method),	494 findunits() (translate.storage.omegat.OmegaTFile method), 499
665 findunit() (translate.storage.tmx.tmxfile method), 670	494 findunits() (translate.storage.omegat.OmegaTFile method), 499
findunit() (translate.storage.tmx.tmxfile method), 670 findunit() (translate.storage.trados.TradosTxtTmFile method), 679 findunit() (translate.storage.ts2.tsfile method), 681	findunits() (translate.storage.omegat.OmegaTFile method), 499 findunits() (translate.storage.omegat.OmegaTFileTab method), 501
findunit() (translate.storage.tmx.tmxfile method), 670 findunit() (translate.storage.trados.TradosTxtTmFile method), 679	494 findunits() (translate.storage.omegat.OmegaTFile method), 499 findunits() (trans- late.storage.omegat.OmegaTFileTab method),
findunit() (translate.storage.tmx.tmxfile method), 670 findunit() (translate.storage.trados.TradosTxtTmFile method), 679 findunit() (translate.storage.ts2.tsfile method), 681 findunit() (translate.storage.txt.TxtFile method),	findunits() (translate.storage.omegat.OmegaTFile method), 499 findunits() (translate.storage.omegat.OmegaTFileTab method), 501 findunits() (translate.storage.php.LaravelPHPFile method), 558
findunit() (translate.storage.tmx.tmxfile method), 670 findunit() (translate.storage.trados.TradosTxtTmFile method), 679 findunit() (translate.storage.ts2.tsfile method), 681 findunit() (translate.storage.txt.TxtFile method), 687 findunit() (translate.storage.utx.UtxFile method),	findunits() (translate.storage.omegat.OmegaTFile method), 499 findunits() (translate.storage.omegat.OmegaTFileTab method), 501 findunits() (translate.storage.php.LaravelPHPFile method), 558 findunits() (translate.storage.php.phpfile method), 563
findunit () (translate.storage.tmx.tmxfile method), 670 findunit () (translate.storage.trados.TradosTxtTmFile method), 679 findunit () (translate.storage.ts2.tsfile method), 681 findunit () (translate.storage.txt.TxtFile method), 687 findunit () (translate.storage.utx.UtxFile method), 692 findunit () (translate.storage.wordfast.WordfastTMFile	findunits() (translate.storage.omegat.OmegaTFile method), 499 findunits() (translate.storage.omegat.OmegaTFileTab method), 501 findunits() (translate.storage.php.LaravelPHPFile method), 558 findunits() (translate.storage.php.phpfile method), 563 findunits() (translate.storage.pocommon.pofile
findunit() (translate.storage.tmx.tmxfile method), 670 findunit() (translate.storage.trados.TradosTxtTmFile method), 679 findunit() (translate.storage.ts2.tsfile method), 681 findunit() (translate.storage.txt.TxtFile method), 687 findunit() (translate.storage.utx.UtxFile method), 692 findunit() (translate.storage.wordfast.WordfastTMFile method), 698 findunit() (translate.storage.xliff.xlifffile method),	findunits() (translate.storage.omegat.OmegaTFile method), 499 findunits() (translate.storage.omegat.OmegaTFileTab method), 501 findunits() (translate.storage.php.LaravelPHPFile method), 558 findunits() (translate.storage.php.phpfile method), 563 findunits() (translate.storage.pocommon.pofile method), 568 findunits() (translate.storage.poxliff.PoXliffFile method), 576 findunits() (translate.storage.properties.gwtfile
findunit() (translate.storage.tmx.tmxfile method), 670 findunit() (translate.storage.trados.TradosTxtTmFile method), 679 findunit() (translate.storage.ts2.tsfile method), 681 findunit() (translate.storage.txt.TxtFile method), 687 findunit() (translate.storage.utx.UtxFile method), 692 findunit() (translate.storage.wordfast.WordfastTMFile method), 698 findunit() (translate.storage.xliff.xlifffile method), 705 findunits() (translate.storage.base.DictStore method), 404 findunits() (translate.storage.base.TranslationStore	findunits () (translate.storage.omegat.OmegaTFile method), 499 findunits () (translate.storage.omegat.OmegaTFileTab method), 501 findunits () (translate.storage.php.LaravelPHPFile method), 558 findunits () (translate.storage.php.phpfile method), 563 findunits () (translate.storage.pocommon.pofile method), 568 findunits () (translate.storage.poxliff.PoXliffFile method), 576 findunits () (translate.storage.properties.gwtfile method), 596 findunits () (translate.storage.properties.javafile
findunit () (translate.storage.tmx.tmxfile method), 670 findunit () (translate.storage.trados.TradosTxtTmFile method), 679 findunit () (translate.storage.ts2.tsfile method), 681 findunit () (translate.storage.txt.TxtFile method), 687 findunit () (translate.storage.utx.UtxFile method), 692 findunit () (translate.storage.wordfast.WordfastTMFile method), 698 findunit () (translate.storage.xliff.xlifffile method), 705 findunits () (translate.storage.base.DictStore method), 404 findunits () (translate.storage.base.TranslationStore method), 409 findunits () (translate.storage.catkeys.CatkeysFile	findunits() (translate.storage.omegat.OmegaTFile method), 499 findunits() (translate.storage.omegat.OmegaTFileTab method), 501 findunits() (translate.storage.php.LaravelPHPFile method), 558 findunits() (translate.storage.php.phpfile method), 563 findunits() (translate.storage.pocommon.pofile method), 568 findunits() (translate.storage.poxliff.PoXliffFile method), 576 findunits() (translate.storage.properties.gwtfile method), 596 findunits() (translate.storage.properties.javafile method), 598 findunits() (translate.storage.properties.javafile method), 598 findunits() (translate.storage.properties.javafile method), 598
findunit() (translate.storage.tmx.tmxfile method), 670 findunit() (translate.storage.trados.TradosTxtTmFile method), 679 findunit() (translate.storage.ts2.tsfile method), 681 findunit() (translate.storage.txt.TxtFile method), 687 findunit() (translate.storage.utx.UtxFile method), 692 findunit() (translate.storage.wordfast.WordfastTMFile method), 698 findunit() (translate.storage.xliff.xlifffile method), 705 findunits() (translate.storage.base.DictStore method), 404 findunits() (translate.storage.base.TranslationStore method), 409	findunits() (translate.storage.omegat.OmegaTFile method), 499 findunits() (translate.storage.omegat.OmegaTFileTab method), 501 findunits() (translate.storage.php.LaravelPHPFile method), 558 findunits() (translate.storage.php.phpfile method), 563 findunits() (translate.storage.pocommon.pofile method), 568 findunits() (translate.storage.pocommon.pofile method), 576 findunits() (translate.storage.pocoliff.PoXliffFile method), 596 findunits() (translate.storage.properties.gwtfile method), 596 findunits() (translate.storage.properties.javafile method), 598
findunit () (translate.storage.tmx.tmxfile method), 670 findunit () (translate.storage.trados.TradosTxtTmFile method), 679 findunit () (translate.storage.ts2.tsfile method), 681 findunit () (translate.storage.txt.TxtFile method), 687 findunit () (translate.storage.utx.UtxFile method), 692 findunit () (translate.storage.wordfast.WordfastTMFile method), 698 findunit () (translate.storage.xliff.xlifffile method), 705 findunits () (translate.storage.base.DictStore method), 404 findunits () (translate.storage.base.TranslationStore method), 409 findunits () (translate.storage.catkeys.CatkeysFile method), 416	findunits () (translate.storage.omegat.OmegaTFile method), 499 findunits () (translate.storage.omegat.OmegaTFileTab method), 501 findunits () (translate.storage.php.LaravelPHPFile method), 558 findunits () (translate.storage.php.phpfile method), 563 findunits () (translate.storage.pocommon.pofile method), 568 findunits () (translate.storage.poxliff.PoXliffFile method), 576 findunits () (translate.storage.properties.gwtfile method), 596 findunits () (translate.storage.properties.javafile method), 598 findunits () (translate.storage.properties.javafile method), 598 findunits () (translate.storage.properties.javafile method), 598
findunit () (translate.storage.tmx.tmxfile method), 670 findunit () (translate.storage.trados.TradosTxtTmFile method), 679 findunit () (translate.storage.ts2.tsfile method), 681 findunit () (translate.storage.txt.TxtFile method), 687 findunit () (translate.storage.utx.UtxFile method), 692 findunit () (translate.storage.wordfast.WordfastTMFile method), 698 findunit () (translate.storage.xliff.xlifffile method), 705 findunits () (translate.storage.base.DictStore method), 404 findunits () (translate.storage.base.TranslationStore method), 409 findunits () (translate.storage.catkeys.CatkeysFile method), 416 findunits () (translate.storage.csvl10n.csvfile	findunits () (translate.storage.omegat.OmegaTFile method), 499 findunits () (translate.storage.omegat.OmegaTFileTab method), 501 findunits () (translate.storage.php.LaravelPHPFile method), 558 findunits () (translate.storage.php.phpfile method), 563 findunits () (translate.storage.pocommon.pofile method), 568 findunits () (translate.storage.poxliff.PoXliffFile method), 576 findunits () (translate.storage.properties.gwtfile method), 596 findunits () (translate.storage.properties.javafile method), 598 findunits () (translate.storage.properties.javafile method), 598 findunits () (translate.storage.properties.javafile method), 600

findunits() (translate.storage.properties.propfile firstChild (translate.misc.ourdom.Document method), 605 *tribute*), 397 findunits() (translate.storage.properties.stringsfile firstChild (translate.misc.ourdom.Element tribute), 397 *method*), 613 findunits() (transflatten() (translate.storage.placeables.base.Bpt method), 509 late.storage.properties.stringsutf8file method), 615 (translate.storage.placeables.base.Bx flatten() findunits() (translate.storage.properties.xwikifile method), 517 method), 617 flatten() (translate.storage.placeables.base.Ept findunits() (transmethod), 511 late. storage. properties. XWikiFull Pageflatten() (translate.storage.placeables.base.Ex *method*), 592 method), 519 flatten() findunits() (translate.storage.placeables.base.G (translate.storage.properties.XWikiPageProperties method), 515 method), 594(translate.storage.placeables.base.It flatten() findunits() (translate.storage.pypo.pofile method), method), 514 (translate.storage.placeables.base.Ph 622 flatten() findunits() (translate.storage.qm.qmfile method), method), 512 629 (translate.storage.placeables.base.Sub flatten() method), 522 findunits () (translate.storage.qph.QphFile method), 634 flatten() (translate.storage.placeables.base.X findunits () (translate.storage.rc.rcfile method), 640 method), 520 findunits() $\verb|flatten()| (translate.storage.placeables.general.AltAttrPlaceable$ (translate.storage.subtitles.AdvSubStationAlphaFilemethod), 523 $\verb|flatten()| (translate.storage.placeables.general.XMLE ntity Placeable$ *method*), 647 findunits() (transmethod), 525 late.storage.subtitles.MicroDVDFile method), $\verb|flatten()| (translate.storage.placeables.general.XMLTagPlaceable$ method), 527 findunits() (translate.storage.subtitles.SubRipFile flatten() (translate.storage.placeables.interfaces.BasePlaceable *method*), 651 method), 529 findunits() (transflatten() (translate.storage.placeables.interfaces.InvisiblePlaceable late.storage.subtitles.SubStationAlphaFile method), 530 *method*), 652 flatten() (translate.storage.placeables.interfaces.MaskingPlaceable findunits() (translate.storage.subtitles.SubtitleFile method), 532 *method*), 654 flatten() (translate.storage.placeables.interfaces.ReplacementPlaceables.ReplacementPlaceables.ReplacementPlaceables.ReplacementPlaceables.ReplacementPlaceables.ReplacementPlaceables.ReplacementPlaceables.ReplacementPlaceables.ReplacementPlaceables.ReplacementPlaceables.ReplacementPlaceables.ReplacementPlaceables.Replaceabl findunits() (translate.storage.tbx.tbxfile method), method), 534 flatten() (translate.storage.placeables.interfaces.SubflowPlaceable findunits()(translate.storage.tiki.TikiStore method), *method*), 535 flatten() (translate.storage.placeables.strelem.StringElem findunits() (translate.storage.tmx.tmxfile method), method), 538 flatten() (translate.storage.placeables.terminology.TerminologyPlacea 670 findunits() (trans*method*), 540 flatten() (translate.storage.placeables.xliff.Bpt late.storage.trados.TradosTxtTmFile method), method), 542 findunits () (translate.storage.ts2.tsfile method), 681 flatten() (translate.storage.placeables.xliff.Bx findunits() (translate.storage.txt.TxtFile method), method), 547 687 flatten() (translate.storage.placeables.xliff.Ept findunits() (translate.storage.utx.UtxFile method), *method*), 543 692 (translate.storage.placeables.xliff.Ex flatten() findunits() method), 548 (translate.storage.placeables.xliff.G late.storage.wordfast.WordfastTMFile method), flatten() method), 550 findunits() (translate.storage.xliff.xlifffile method), flatten() (translate.storage.placeables.xliff.It

772 Index

method), 551

705

flatter	n () (translate.storage.placeabl	es.xliff.Ph	metho	d), 393	
	method), 555		fr (class in tran	nslate.lang.fr), 369	
flatter	n() (translate.storage.placeable method), 553	s.xliff.Sub	fromkeys() 389	(translate.misc.dictutils.cidic	t method),
flatter	n () (translate.storage.placeables.xliff. method), 556	UnknownX	_	(translate.storage.oo.unord), 507	malizechar
flatter		oles.xliff.X		unslate.lang.common.Common	attribute),
flatter	n () (translate.tools.poconflicts.Conflic method), 716	ctOptionPar		(in module translate.filters.he) (in module translate.filte	_
_	aia_plurals() late.convert.prop2po.prop2po method	(trans-	352 functions()		(trans-
fold_gv	vt_plurals()	(trans-		ters.checks.CCLicenseChecke	`
	late.convert.prop2po.prop2po method nicode() (in module translate.lang.			(translate filters checks Dru	nalChackar
format				d), 281	риспескет
	method), 390	unnsming		a), 201 (translate.filters.checks.Gno	meChecker
	_manpage()	(trans-		d), 287	теспескет
	late.convert.convert.ArchiveConvertO method), 238	,	functions()	(translate.filters.checks.l d), 292	OSChecker
format_	_manpage() late.convert.convertOptionPa	(trans-	<pre>functions()</pre>		KdeChecker
	method), 242		functions()		20nChecker
	_manpage()	(trans-		d), 304	
	late.convert.po2moz.MozConvertOpti	onParser	functions()	**	(trans-
	method), 254			ters.checks.LibreOfficeChecke	,
format_	_manpage()	(trans-	310		
	late.convert.po2tmx.TmxOptionParset	r	functions()		(trans-
	method), 259		late.fil	ters.checks.MinimalChecker	method),
format_	_manpage()	(trans-	315		
	late.convert.po2wordfast.WfOptionPamethod), 264	rser		(translate.filters.checks.Moz d), 321	illaChecker
format_	_manpage()	(trans-	functions()		(trans-
	late.filters.pofilter.FilterOptionParser 353	method),	late.fil 327	ters.checks.OpenOfficeChecke	r method),
format_	_manpage()	(trans-	functions()		(trans-
	late.misc.optrecurse.RecursiveOption method), 394	Parser	late.fil 333	ters.checks.ReducedChecker	method),
	_manpage()	(trans-	functions()		(trans-
	late.tools.poconflicts.ConflictOptionP method), 716	arser	late.fil 338	ters.checks.StandardChecker	method),
format_	_manpage()	(trans-	functions()	(translate.filters.checks.Te	rmChecker
	late.tools.pogrep.GrepOptionParser	method),	metho	d), 346	
	720		fuzzy_unito	count()	(trans-
	_manpage() late.tools.porestructure.SplitOptionPa	(trans- irser	645	orage.statistics.Statistics	method),
	method), 723			() (translate.storage.statistic	cs.Statistics
	manpage() late.tools.poterminology.Terminology	(trans- OptionParse	er	d), 645	
	method), 727		G		
	_map() (translate.misc.multistring.n method), 390			late.storage.placeables.base), late.storage.placeables.xliff),	
	_option_strings() <i>late.misc.optrecurse.ManHelpFormat</i> .	(trans- ter	gconf()	(translate.filters.checks.Gno d), 287	

<pre>generate_dialog_caption_name() (in module</pre>	<pre>get_ignored_filters() (trans-</pre>
translate.storage.rc), 639	$late.filters.checks.LibreOfficeChecker\ method),$
<pre>generate_dialog_control_name() (in module</pre>	310
translate.storage.rc), 639	<pre>get_ignored_filters() (trans-</pre>
<pre>generate_menu_pre_name() (in module trans-</pre>	late.filters.checks.MinimalChecker method),
late.storage.rc), 639	316
<pre>generate_menuitem_name() (in module trans-</pre>	<pre>get_ignored_filters() (trans-</pre>
late.storage.rc), 639	late.filters.checks.MozillaChecker method),
<pre>generate_popup_caption_name() (in module</pre>	321
translate.storage.rc), 639	<pre>get_ignored_filters() (trans-</pre>
<pre>generate_popup_pre_name() (in module trans- late.storage.rc), 639</pre>	late.filters.checks.OpenOfficeChecker method), 327
<pre>generate_stringtable_name() (in module</pre>	<pre>get_ignored_filters() (trans-</pre>
translate.storage.rc), 640	late.filters.checks.ReducedChecker method),
get () (translate.misc.dictutils.cidict method), 389	333
get() (translate.storage.oo.unormalizechar method),	<pre>get_ignored_filters() (trans-</pre>
507	late.filters.checks.StandardChecker method),
get_abs_data_filename() (in module trans-	339
late.misc.file_discovery), 389	get_ignored_filters() (trans-
get_all_languages() (in module trans-	late.filters.checks.StandardUnitChecker
	method), 342
late.lang.factory), 368	
get_country_iso_name() (in module trans-	
late.lang.data), 364	late.filters.checks.TermChecker method),
<pre>get_file() (translate.storage.bundleprojstore.BundleP</pre>	
method), 414	<pre>get_ignored_filters() (trans-</pre>
<pre>get_file() (translate.storage.project.Project</pre>	late.filters.checks.TranslationChecker method), 349
<pre>get_file() (translate.storage.projstore.ProjectStore</pre>	
method), 584	late.filters.checks.UnitChecker method),
<pre>get_filename_type() (trans-</pre>	350
late.storage.bundleprojstore.BundleProjectStore	<pre>get_index_data() (trans-</pre>
method), 414	late.storage.placeables.base.Bpt method),
<pre>get_filename_type() (trans-</pre>	509
	<pre>get_index_data() (trans-</pre>
584	late.storage.placeables.base.Bx method),
<pre>get_from_lines() (translate.tools.pydiff.FileDiffer</pre>	517
method), 730	<pre>get_index_data() (trans-</pre>
get_ignored_filters() (trans-	late.storage.placeables.base.Ept method),
late.filters.checks.CCLicenseChecker method),	511
275	
	get_index_data() (trans-
<pre>get_ignored_filters() (trans-</pre>	late.storage.placeables.base.Ex method),
late.filters.checks.DrupalChecker method),	519
281	get_index_data() (trans-
<pre>get_ignored_filters() (trans-</pre>	late.storage.placeables.base.G method),
late.filters.checks.GnomeChecker method),	515
287	get_index_data() (trans-
<pre>get_ignored_filters() (trans-</pre>	late.storage.placeables.base.It method),
late.filters.checks.IOSChecker method), 293	514
<pre>get_ignored_filters() (trans-</pre>	<pre>get_index_data()</pre> (trans-
late.filters.checks.KdeChecker method),	late.storage.placeables.base.Ph method),
298	512
<pre>get_ignored_filters() (trans-</pre>	<pre>get_index_data() (trans-</pre>
late.filters.checks.L20nChecker method),	late.storage.placeables.base.Sub method),
304	522

get_index_data()	(trans-	553	
late.storage.placeables.base.X	method),	<pre>get_index_data()</pre>	(trans-
520		late.storage.placeables.xliff.Unknov	vnXML
<pre>get_index_data()</pre>	(trans-	method), 556	
late. storage. placeables. general. Alt Anti-Anti-Anti-Anti-Anti-Anti-Anti-Anti-	ttrPlaceable		(trans-
method), 524		late.storage.placeables.xliff.X meth	
<pre>get_index_data()</pre>	(trans-		n module
late.storage.placeables.general.XML	EntityPlace		1.1
method), 525	(trans-	<pre>get_language_iso_name() (in mod late.lang.data), 364</pre>	dule trans-
<pre>get_index_data() late.storage.placeables.general.XML</pre>	(0	(trans-
method), 527	nagi iaccao	late.storage.properties.propunit cla	`
get_index_data()	(trans-	610	iss memou),
late.storage.placeables.interfaces.Ba	sePlaceable	get_missing_part()	(trans-
method), 529		late.storage.properties.xwikiunit cla	uss method),
<pre>get_index_data()</pre>	(trans-	619	
late.storage.placeables.interfaces.Inv	visiblePlace	$ab\hspace{-0.1cm}/\hspace{-0.1cm}/e$ t_parent_elem()	(trans-
method), 530		late.storage.placeables.base.Bpt	method),
<pre>get_index_data()</pre>	(trans-	509	,
late.storage.placeables.interfaces.Mo	askingPlace		(trans-
method), 532	(4	late.storage.placeables.base.Bx	method),
<pre>get_index_data() late.storage.placeables.interfaces.Re</pre>	(trans-	517	(tuans
method), 534	рисетепіғ	late.storage.placeables.base.Ept	(trans- method),
get_index_data()	(trans-	511	memoa),
late.storage.placeables.interfaces.Su	((trans-
method), 536	J	late.storage.placeables.base.Ex	method),
<pre>get_index_data()</pre>	(trans-	519	
late.storage.placeables.strelem.String	gElem	<pre>get_parent_elem()</pre>	(trans-
method), 538		late.storage.placeables.base.G	method),
<pre>get_index_data()</pre>	(trans-	516	
late.storage.placeables.terminology.	Terminology		(trans-
method), 540	(tuana	late.storage.placeables.base.It 514	method),
get_index_data()	(trans- method),		(trans-
tate.storage.ptaceabtes.xttyj.Бpt 542	memoa),	<pre>get_parent_elem() late.storage.placeables.base.Ph</pre>	method),
get_index_data()	(trans-	512	memoa),
late.storage.placeables.xliff.Bx	method),	<pre>get_parent_elem()</pre>	(trans-
547	,,	late.storage.placeables.base.Sub	method),
<pre>get_index_data()</pre>	(trans-	522	
late.storage.placeables.xliff.Ept	method),	<pre>get_parent_elem()</pre>	(trans-
543		late.storage.placeables.base.X	method),
<pre>get_index_data()</pre>	(trans-	520	
late.storage.placeables.xliff.Ex	method),	<pre>get_parent_elem()</pre>	(trans-
548	(4	late.storage.placeables.general.Alt	AttrPlaceable
get_index_data()	(trans-	method), 524	(tuans
<pre>late.storage.placeables.xliff.G metho get_index_data()</pre>	a), 550 (trans-	<pre>get_parent_elem() late.storage.placeables.general.XM</pre>	(trans- I EntityPlaceable
late.storage.placeables.xliff.It method	`	method), 525	LLmiiyi iaceabie
<pre>get_index_data()</pre>	(trans-	get_parent_elem()	(trans-
late.storage.placeables.xliff.Ph	method),	late.storage.placeables.general.XM	,
555	,,	method), 527	-
<pre>get_index_data()</pre>	(trans-	<pre>get_parent_elem()</pre>	(trans-
late.storage.placeables.xliff.Sub	method),	late.storage.placeables.interfaces.B	BasePlaceable

method), 529		<pre>get_real_filename()</pre>	(trans-
get_parent_elem()	(trans-	late.storage.project.Project method)	, 583
late.storage.placeables.interfaces.l	Invisible Place G	a lde t_rich_target()	(trans-
method), 531		late.storage.poxliff.PoXliffUnit	method),
<pre>get_parent_elem()</pre>	(trans-	580	
late.storage.placeables.interfaces.l method), 532	MaskingPlace	<pre>abjet_rich_target() (translate.storage.x</pre>	liff.xliffunit
<pre>get_parent_elem()</pre>	(trans-	get_source_text()	(trans-
	ReplacementP	laceable late.storage.statistics.Statistics 645	method),
<pre>get_parent_elem()</pre>	(trans-	<pre>get_starttag_text()</pre>	(trans-
late.storage.placeables.interfaces. State of the state		able late.storage.html.htmlfile method), 4	36
method), 536		<pre>get_starttag_text()</pre>	(trans-
get_parent_elem()	(trans-	late.storage.html.POHTMLParser	method),
late.storage.placeables.strelem.Str method), 538	ingElem	434 get_time() (translate.storage.trados.Tra	dosTxtDate
<pre>get_parent_elem()</pre>	(trans-	method), 675	
	•	Physiceablisme () (translate.storage.wordfast.Wo method), 700	ordfastTime
get_parent_elem()	(trans-	<pre>get_timestring()</pre>	(trans-
late.storage.placeables.xliff.Bpt 542	method),	late.storage.trados.TradosTxtDate 675	method),
<pre>get_parent_elem()</pre>	(trans-	<pre>get_timestring()</pre>	(trans-
late.storage.placeables.xliff.Bx 547	method),	late.storage.wordfast.WordfastTime	method),
<pre>get_parent_elem()</pre>	(trans-	<pre>get_to_lines() (translate.tools.pydij</pre>	ff.FileDiffer
late.storage.placeables.xliff.Ept 544	method),	<pre>method),730 get_xliff_source_target_doms()</pre>	
get_parent_elem()	(trans-	translate.storage.xml_extract.genero	
late.storage.placeables.xliff.Ex	method),	getaccelerators() (in module	
548	memou),	late.filters.decoration), 351	iians
get_parent_elem()	(trans-	getalttrans() (translate.storage.poxliff.	PoXliffI Init
late.storage.placeables.xliff.G meth	,	method), 580	1 oznaj e ma
<pre>get_parent_elem()</pre>	(trans-	getalttrans() (translate.storage.p	ovpo pounit
late.storage.placeables.xliff.It meth		method), 625	ээрогрошиг
<pre>get_parent_elem()</pre>	(trans-	getalttrans() (translate.storage.x	cliff.xliffunit
late.storage.placeables.xliff.Ph	method),	method), 708	33 33
555	,,	getarchiveclass()	(trans-
get_parent_elem()	(trans-	late.convert.convert.ArchiveConvert	OptionParse.
late.storage.placeables.xliff.Sub	method),	method), 238	-
553		getarchiveclass()	(trans-
<pre>get_parent_elem()</pre>	(trans-	late.convert.po2tmx.TmxOptionPars	er
late.storage.placeables.xliff.Unkno	wnXML	method), 259	
method), 557		getarchiveclass()	(trans-
get_parent_elem()	(trans-	late.convert.po 2 word fast. Wf Option F	Parser
late.storage.placeables.xliff.X meth	od), 545	method), 264	
get_proj_filename()	(trans-	getautomaticcomments()	(trans-
late.storage.bundleprojstore.Bundle method), 414	eProjectStore	late.storage.poxliff.PoXliffUnit 580	method),
get_proj_filename()	(trans-	getbodynode() (translate.storage.poxliff	PoXliffFile
late.storage.project.Project method	1), 583	method), 576	**
get_proj_filename()	(trans-	getbodynode() (translate.storage.	xliff.xlifffile
late.storage.projstore.ProjectStore	method),	method), 705	
584		getclass() (in module translate.storage.fa	ctory), 432

getcontext() (translate.storage.base.DictUnit method), 406	late.storage.properties.proppluralunit method), 607
getcontext() (trans-	<pre>getcontext() (translate.storage.properties.propunit</pre>
late.storage.base.TranslationUnit method),	method), 610
411	getcontext() (translate.storage.properties.xwikiunit
getcontext() (translate.storage.catkeys.CatkeysUnit method), 418	method), 619 getcontext() (translate.storage.pypo.pounit
getcontext() (translate.storage.csvl10n.csvunit method), 423	<pre>method), 625 getcontext() (translate.storage.qm.qmunit method),</pre>
getcontext() (translate.storage.dtd.dtdunit method),	631
429	getcontext() (translate.storage.qph.QphUnit
getcontext() (translate.storage.html.htmlunit	method), 636
method), 438	getcontext() (translate.storage.rc.rcunit method),
getcontext() (translate.storage.ical.icalunit	642
method), 444	getcontext() (trans-
getcontext() (translate.storage.ini.iniunit method),	late.storage.subtitles.SubtitleUnit method),
449	656
getcontext() (trans-	<pre>getcontext() (translate.storage.tbx.tbxunit method),</pre>
late.storage.jsonl10n.ARBJsonUnit method),	661
454	getcontext() (translate.storage.tiki.TikiUnit
getcontext() (trans-	method), 667
late.storage.jsonl10 n. GoI18 NJ son Unit	getcontext() (translate.storage.tmx.tmxunit
method), 459	method), 672
getcontext() (trans-	<pre>getcontext() (translate.storage.trados.TradosUnit</pre>
late.storage.jsonl10n.I18NextUnit method),	method), 676
464	<pre>getcontext() (translate.storage.ts2.tsunit method),</pre>
getcontext() (trans-	683
late.storage.jsonl10n.JsonNestedUnit method), 471	getcontext() (translate.storage.txt.TxtUnit method), 689
getcontext() (translate.storage.jsonl10n.JsonUnit method), 474	getcontext() (translate.storage.utx.UtxUnit method), 694
getcontext() (trans-	getcontext() (trans-
late.storage.jsonl10n.WebExtensionJsonUnit	late.storage.wordfast.WordfastUnit method),
method), 479	701
getcontext() (translate.storage.lisa.LISAunit method), 484	getcontext() (translate.storage.xliff.xliffunit method), 708
<pre>getcontext() (translate.storage.mo.mounit method),</pre>	getcontextgroups() (trans-
491	$late.storage.pox liff. PoX liff Unit \\ method),$
getcontext() (trans-	580
late.storage.mozilla_lang.LangUnit method),	getcontextgroups() (trans-
496	late.storage.xliff.xliffunit method), 708
getcontext() (trans-	<pre>getdatatype() (translate.storage.poxliff.PoXliffFile</pre>
late.storage.omegat.OmegaTUnit method),	method), 576
503	getdatatype() (translate.storage.xliff.xlifffile
getcontext() (trans-	method), 705
late.storage.php.LaravelPHPUnit method), 560	getdate() (translate.storage.poxliff.PoXliffFile method), 577
getcontext() (translate.storage.php.phpunit	getdate() (translate.storage.xliff.xlifffile method), 705
method), 565	getdict() (translate.storage.catkeys.CatkeysUnit
<pre>getcontext() (translate.storage.pocommon.pounit</pre>	method), 418
method), 571	getdict() (translate.storage.omegat.OmegaTUnit
getcontext() (translate.storage.poxliff.PoXliffUnit	method), 503
method), 580	getdict() (translate.storage.utx.UtxUnit method),
getcontext() (trans-	694

- (translate.storage.wordfast.WordfastUnit getdict() method), 701 getElementsByTagName helper() (in module translate.misc.ourdom), 397 getemails () (in module translate.filters.decoration), geterrors() (translate.storage.base.DictUnit method), 406 geterrors() (translate.storage.base.TranslationUnit method), 411 geterrors() (translate.storage.catkeys.CatkeysUnit *method*), 418 (translate.storage.csvl10n.csvunit geterrors() method), 423 geterrors() (translate.storage.dtd.dtdunit method), 429 (translate.storage.html.htmlunit geterrors() method), 438 geterrors() (translate.storage.ical.icalunit method), geterrors() (translate.storage.ini.iniunit method), 449 (transgeterrors() late.storage.jsonl10n.ARBJsonUnit method), 455 geterrors() (translate.storage.jsonl10n.GoI18NJsonUnit *method*), 459 geterrors() (translate.storage.jsonl10n.I18NextUnit method), 464geterrors() (translate.storage.jsonl10n.JsonNestedUnit method), 471 (translate.storage.jsonl10n.JsonUnit geterrors() *method*), 474 geterrors() (translate.storage.jsonl10n.WebExtensionJsonUnit method), 479 (translate.storage.lisa.LISAunit geterrors() method), 484 geterrors() (translate.storage.mo.mounit method), 491 geterrors() (translate.storage.mozilla_lang.LangUnit method), geterrors() (translate.storage.omegat.OmegaTUnit *method*), 503 geterrors() (translate.storage.php.LaravelPHPUnit method), 560 geterrors() (translate.storage.php.phpunit method), 565 geterrors() (translate.storage.pocommon.pounit method), 571 (translate.storage.poxliff.PoXliffUnit geterrors()
- method), 580 (transgeterrors() late.storage.properties.proppluralunit method), geterrors() (translate.storage.properties.propunit method), 610 geterrors() (translate.storage.properties.xwikiunit method), 619 geterrors() (translate.storage.pypo.pounit method), 625 geterrors() (translate.storage.qm.qmunit method), 631 (translate.storage.qph.QphUnit geterrors() *method*), 636 geterrors () (translate.storage.rc.rcunit method), 642 geterrors() (translate.storage.subtitles.SubtitleUnit *method*), 656 geterrors() (translate.storage.tbx.tbxunit method), 661 geterrors () (translate.storage.tiki.TikiUnit method), 667 geterrors() (translate.storage.tmx.tmxunit method), 672 (translate.storage.trados.TradosUnit geterrors() method), 676 geterrors() (translate.storage.ts2.tsunit method), 683 geterrors() (translate.storage.txt.TxtUnit method), 689 geterrors() (translate.storage.utx.UtxUnit method), 694 geterrors() (translate.storage.wordfast.WordfastUnit method), geterrors() (translate.storage.xliff.xliffunit method), 708 getfilename() (translate.storage.poxliff.PoXliffFile method), 577 getfilename() (translate.storage.xliff.xlifffile method), 705 getfilenames()(translate.storage.poxliff.PoXliffFile method), 577 getfilenames() (translate.storage.xliff.xlifffile method), 705 getfilenode() (translate.storage.poxliff.PoXliffFile method), 577 getfilenode() (translate.storage.xliff.xlifffile method), 705 (translate.storage.directory.Directory getfiles() method), 426 getfiles() (translate.storage.zip.ZIPFile method), 714

(trans-

late.filters.checks.CCLicenseChecker method),

778 Index

getfilters()

275		method), 259
<pre>getfilters()</pre>	rans- g	etformathelp() (trans-
late.filters.checks.DrupalChecker meti 281	(hod),	late.convert.po2wordfast.WfOptionParser method), 264
getfilters() (to	<i>rans-</i> g	etformathelp() (trans-
late.filters.checks.GnomeChecker meth 287	hod),	late.filters.pofilter.FilterOptionParser method), 353
$\verb getfilters() & \textit{(translate.filters.checks.IOSChe} $	<i>ecker</i> g	etformathelp() (trans-
<pre>method), 293 getfilters() (translate.filters.checks.KdeChe</pre>	ecker	late.misc.optrecurse.RecursiveOptionParser method), 394
method), 298	g	etformathelp() (trans-
<pre>getfilters() (translate.filters.checks.L20nChe</pre>	ecker	$late. tools. poconflicts. Conflict Option Parser\\ method), 716$
· · · · · · · · · · · · · · · · · · ·		etformathelp() (trans-
late.filters.checks.LibreOfficeChecker meth 310	(hod),	late.tools.pogrep.GrepOptionParser method), 720
· ·		etformathelp() (trans-
late.filters.checks.MinimalChecker met 316	(hod),	late.tools.porestructure.SplitOptionParser method), 723
· ·		etformathelp() (trans-
321	(hod),	late.tools.poterminology.TerminologyOptionParsamethod), 727
· · · · · · · · · · · · · · · · · · ·		etfullinputpath() (trans-
late.filters.checks.OpenOfficeChecker meth 327		late.convert.convert.ArchiveConvertOptionParser method), 238
		etfullinputpath() (trans-
late.filters.checks.ReducedChecker meth 333	(hod),	late.convert.convert.ConvertOptionParser method), 242
•		etfullinputpath() (trans-
late.filters.checks.StandardChecker meth 339	(hod),	late.convert.po2moz.MozConvertOptionParser method), 254
· ·	<i>rans-</i> g	etfullinputpath() (trans-
late.filters.checks.StandardUnitChecker method), 342		late.convert.po2tmx.TmxOptionParser method), 259
getfilters() (translate.filters.checks.TeeChe	<i>ecker</i> g	
method), 343	7	late.convert.po2wordfast.WfOptionParser
getfilters() (translate.filters.checks.TermChe		method), 264
<pre>method), 346 getfilters() (ti</pre>	rans-	<pre>etfullinputpath() (trans- late.filters.pofilter.FilterOptionParser method),</pre>
late.filters.checks.TranslationChecker meth		353
350		etfullinputpath() (trans-
<pre>getfilters() (translate.filters.checks.UnitChe method), 350</pre>	_	late.misc.optrecurse.RecursiveOptionParser method), 394
	rans- g	etfullinputpath() (trans-
late.convert.convert.ArchiveConvertOption method), 238	nParser	late.tools.poconflicts.ConflictOptionParser method), 716
<pre>getformathelp()</pre>	rans- g	etfullinputpath() (trans-
late.convert.convert.ConvertOptionParser method), 242		late.tools.pogrep.GrepOptionParser method), 720
	rans- g	etfullinputpath() (trans-
late.convert.po2moz.MozConvertOptionPa method), 254	ırser	late.tools.porestructure.SplitOptionParser method), 723
± ···	rans- g	etfullinputpath() (trans-
late.convert.po2tmx.TmxOptionParser		late.tools.poterminology.TerminologyOptionPars

	method), 727		metho	od), 394		
getful:	loutputpath()	(trans-	getfulltem	platepath	n ()	(trans-
	late.convert.convert.ArchiveConvertOpt method), 238	ionParser		tools.poconfli od), 716	cts.ConflictOptionPo	arser
getfuli	loutputpath()	(trans-	getfulltem	* *	n ()	(trans-
5	late.convert.convert.ConvertOptionPars method), 242				GrepOptionParser	
getful:	loutputpath()	(trans-	getfulltem	platepath	n ()	(trans-
	late.convert.po2moz.MozConvertOption method), 254	Parser		tools.porestru od), 723	ucture.SplitOptionPa	rser
getful:	loutputpath()	(trans-	getfulltem	platepath	n ()	(trans-
	late.convert.po2tmx.TmxOptionParser method), 259			tools.potermi od), 727	nology.Terminology(OptionParser
getful:	loutputpath()	(trans-	getfunction	ns()	(in module	trans-
	late.convert.po2wordfast.WfOptionPars method), 264	er	<i>late.fi</i> getheader(filters.decora)	tion), 351	(trans-
getful:	loutputpath()	(trans-	late.s	storage.wordj	fast.WordfastHeader	method),
	$late.filters.pofilter.FilterOptionParser\ m$	iethod),	698			
	353		getheadern			(trans-
getful	loutputpath()	(trans-			f.PoXliffFile method	
	late.misc.optrecurse.RecursiveOptionPamethod), 394	rser	getheadern methe	ode () od), 705	(translate.storage.xl	iff.xlifffile
getful:	loutputpath()		getheaderp		(translate.storage.	mo.mofile
	late.tools.poconflicts.ConflictOptionPar method), 716	ser	<i>metho</i> getheaderp	od),488 lural()		(trans-
getful	loutputpath()	(trans-			nmon.pofile method), 568
	late.tools.pogrep.GrepOptionParser n	iethod),				(trans-
	720	,		storage.pohed	ader.poheader	method),
getful.	loutputpath()	(trans-	574	7 7 ()		
	late.tools.porestructure.SplitOptionPars method), 723			storage.poxlij	f.PoXliffFile method	
getful:	loutputpath()				(translate.storage.p	ypo.pofile
	late.tools.poterminology.TerminologyOpmethod), 727		getid()(tran		e.base.DictUnit met	
getful.	ltemplatepath()		getid()		.storage.base.Transl	ationUnit
	late.convert.convert.ArchiveConvertOpt			od), 411		.7
	method), 238				e.storage.catkeys.Ca	tkeysUnit
getful.	ltemplatepath()	(trans-		od), 418		mathad)
	late.convert.convert.ConvertOptionPars method), 242		423		ge.csvl10n.csvunit	
getful.	ltemplatepath()	(trans-			e.dtd.dtdunit method	
	late.convert.po2moz.MozConvertOption method), 254		getid() (tran	nslate.storag	e.html.htmlunit meth e.ical.icalunit metho	d), 444
getful:	ltemplatepath()	(trans-		_	e.ini.iniunit method)	
	late.convert.po2tmx.TmxOptionParser method), 259			od), 455	torage.jsonl10n.ARE	
getful:	ltemplatepath()	(trans-			ige.jsonl10n.GoI18N	IJsonUnit
	late.convert.po2wordfast.WfOptionPars	er		od), 460		
	method), 264		getid()		storage.jsonl10n.I18	SNextUnit
getful:	ltemplatepath()	(trans-		(od), 464		
	late.filters.pofilter.FilterOptionParser n 353		metho	od), 471	rage.jsonl10n.JsonN	
getful:	ltemplatepath() late.misc.optrecurse.RecursiveOptionPa	(trans- rser	getid() (<i>tran</i> 474	nslate.storag	e.jsonl10n.JsonUnit	method),

getid() (translate.storage.jsonl10n.WebExtensionJsonUnitetids() (translate.storage.jsonl10n.GoI18NJsonFile method), 479 getid() (translate.storage.lisa.LISAunit method), 484 getid() (translate.storage.mo.mounit method), 491 getid() (translate.storage.mozilla lang.LangUnit (translate.storage.omegat.OmegaTUnit getid() *method*), 503 getid() (translate.storage.php.LaravelPHPUnit method), 561 getid() (translate.storage.php.phpunit method), 565 getid() (translate.storage.pocommon.pounit method), getid() (translate.storage.poxliff.PoXliffUnit method), 580 getid() (translate.storage.properties.proppluralunit *method*), 607 (translate.storage.properties.propunit getid() *method*), 610 getid() (translate.storage.properties.xwikiunit method), 619 getid() (translate.storage.pypo.pounit method), 625 getid() (translate.storage.qm.qmunit method), 631 getid() (translate.storage.qph.QphUnit method), 637 getid() (translate.storage.rc.rcunit method), 642 getid() (translate.storage.subtitles.SubtitleUnit method), 656 getid() (translate.storage.tbx.tbxunit method), 661 getid() (translate.storage.tiki.TikiUnit method), 667 getid() (translate.storage.tmx.tmxunit method), 672 getid() (translate.storage.trados.TradosUnit method), 676 getid() (translate.storage.ts2.tsunit method), 684 getid() (translate.storage.txt.TxtUnit method), 689 getid() (translate.storage.utx.UtxUnit method), 695 getid() (translate.storage.wordfast.WordfastUnit

method), 701

method), 409

method), 416

method), 434

method), 453

404

421

getids()

getids()

getids()

getids()

getid() (translate.storage.xliff.xliffunit method), 708

getids() (translate.storage.dtd.dtdfile method), 428

getids() (translate.storage.ical.icalfile method), 442

(translate.storage.jsonl10n.ARBJsonFile

getids () (translate.storage.ini.inifile method), 447

method), 457 getids() (translate.storage.jsonl10n.I18NextFile method), 462 getids() (translate.storage.jsonl10n.JsonFile method), 467 (translate.storage.jsonl10n.JsonNestedFile getids() method), 469 getids()(translate.storage.jsonl10n.WebExtensionJsonFile method), 477 getids() (translate.storage.lisa.LISAfile method), 482 getids () (translate.storage.mo.mofile method), 488 (translate.storage.mozilla_lang.LangStore getids() method), 494 (translate.storage.omegat.OmegaTFilegetids() method), 499 (translate.storage.omegat.OmegaTFileTab getids() method), 501 (translate.storage.php.LaravelPHPFile qetids() method), 559 getids() (translate.storage.php.phpfile method), 563 getids()(translate.storage.pocommon.pofile method), 568 (translate.storage.poxliff.PoXliffFile getids() method), 577 getids() (translate.storage.properties.gwtfile method), 596 (translate.storage.properties.javafile getids() method), 598 getids() (translate.storage.properties.javautf16file method), 600 getids() (translate.storage.properties.javautf8file method), 602 (translate.storage.properties.joomlafile getids() method), 604 (translate.storage.properties.propfile getids() *method*), 605 (translate.storage.properties.stringsfile getids() getids() (translate.storage.base.DictStore method), method), 613 (translate.storage.properties.stringsutf8file getids() (translate.storage.base.TranslationStore method), 615 (translate.storage.properties.xwikifile getids() (translate.storage.catkeys.CatkeysFile method), 617 getids() (translate.storage.properties.XWikiFullPage getids() (translate.storage.csvl10n.csvfile method), method), 592 getids()(translate.storage.properties.XWikiPageProperties method), 594 getids() (translate.storage.html.htmlfile method), 436 getids () (translate.storage.pypo.pofile method), 622 (translate.storage.html.POHTMLParser getids() (translate.storage.qm.qmfile method), 629 getids () (translate.storage.qph.QphFile method), 634 getids()(translate.storage.rc.rcfile method), 640

getids()(translate.storage.subtitles.AdvSubStationAlphaFile

(translate.storage.subtitles.MicroDVDFile

method), 647

Index 781

getids()

method), 649	411
<pre>getids() (translate.storage.subtitles.SubRipFile</pre>	getlocations() (trans-
method), 651	late.storage.catkeys.CatkeysUnit method),
$\verb"getids" () \textit{ (translate.storage.subtitles.SubStationAlphaFi}$	<i>le</i> 418
method), 652	<pre>getlocations() (translate.storage.csvl10n.csvunit</pre>
${\tt getids()} \qquad \textit{(translate.storage.subtitles.SubtitleFile)}$	method), 423
method), 654	getlocations() (translate.storage.dtd.dtdunit
getids()(translate.storage.tbx.tbxfile method), 659	method), 430
<pre>getids() (translate.storage.tiki.TikiStore method), 665</pre>	getlocations() (translate.storage.html.htmlunit
getids()(translate.storage.tmx.tmxfile method), 670	method), 439
<pre>getids() (translate.storage.trados.TradosTxtTmFile</pre>	getlocations() (translate.storage.ical.icalunit
method), 679	method), 444
getids() (translate.storage.ts2.tsfile method), 681	getlocations() (translate.storage.ini.iniunit
getids() (translate.storage.txt.TxtFile method), 687	method), 449
getids() (translate.storage.utx.UtxFile method), 692	getlocations() (trans-
<pre>getids() (translate.storage.wordfast.WordfastTMFile</pre>	late.storage.jsonl10n.ARBJsonUnit method),
method), 699	455
getids() (translate.storage.xliff.xlifffile method), 705	getlocations() (trans-
getkey() (translate.storage.oo.ooline method), 506	late.storage.jsonl10n.GoI18NJsonUnit
getlanguage (in module translate.lang.factory), 368	method), 460
getlanguageNode() (translate.storage.lisa.LISAunit	getlocations() (trans-
method), 485	late.storage.jsonl10n.I18NextUnit method),
getlanguageNode() (trans-	465
late.storage.poxliff.PoXliffUnit method),	getlocations() (trans-
580	late.storage.jsonl10n.JsonNestedUnit method),
getlanguageNode() (translate.storage.qph.QphUnit	471
method), 637	getlocations() (trans-
getlanguageNode() (translate.storage.tbx.tbxunit	late.storage.jsonl10n.JsonUnit method),
method), 662	474
getlanguageNode() (translate.storage.tmx.tmxunit	
method), 672	late.storage.jsonl10n.WebExtensionJsonUnit
getlanguageNode() (translate.storage.ts2.tsunit	method), 479
method), 684	getlocations() (translate.storage.lisa.LISAunit
getlanguageNode() (translate.storage.xliff.xliffunit	method), 485
method), 708	getlocations() (translate.storage.mo.mounit
getlanguageNodes() (trans-	method), 491
late.storage.lisa.LISAunit method), 485	getlocations() (trans-
getlanguageNodes() (trans-	
late.storage.poxliff.PoXliffUnit method),	496
581	getlocations() (trans-
getlanguageNodes() (trans-	late.storage.omegat.OmegaTUnit method),
late.storage.qph.QphUnit method), 637	503
getlanguageNodes() (translate.storage.tbx.tbxunit	getlocations() (trans-
method), 662	late.storage.php.LaravelPHPUnit method),
getlanguageNodes() (trans-	561
late.storage.tmx.tmxunit method), 672	getlocations() (translate.storage.php.phpunit
getlanguageNodes() (translate.storage.ts2.tsunit	method), 566
method), 684	getlocations() (trans-
getlanguageNodes() (trans-	late.storage.pocommon.pounit method),
late.storage.xliff.xliffunit method), 708	571
getlocations() (translate.storage.base.DictUnit	getlocations() (trans-
method), 406	late.storage.poxliff.PoXliffUnit method),
getlocations() (trans-	581
late.storage.base.TranslationUnit $method),$	getlocations() (trans-

late.storage.properties.proppluralunit method), 608		method), 708 getnotes() (translate.storage.base.DictUnit method),
getlocations()	(trans-	406
_	perties.propunit method),	getnotes() (translate.storage.base.TranslationUnit
610		method), 412
getlocations()	(trans-	getnotes() (translate.storage.catkeys.CatkeysUnit
	perties.xwikiunit method),	method), 418
619		getnotes() (translate.storage.csvl10n.csvunit
getlocations()	(translate.storage.pypo.pounit	method), 424
method), 625		getnotes() (translate.storage.dtd.dtdunit method),
getlocations()	(translate.storage.qm.qmunit	430
method), 631	(40 months of a month of hills in	getnotes() (translate.storage.html.htmlunit method), 439
<pre>getlocations() method), 637</pre>	(translate.storage.qph.QphUnit	
getlocations()	(translate.storage.rc.rcunit	getnotes() (translate.storage.ical.icalunit method), 444
method), 643	(transtate.storage.rc.rcumi	getnotes() (translate.storage.ini.iniunit method), 449
getlocations()	(trans-	getnotes() (translate.storage.jsonl10n.ARBJsonUnit
_	titles.SubtitleUnit method),	method), 455
656	memou),	getnotes() (translate.storage.jsonl10n.GoI18NJsonUnit
getlocations()	(translate.storage.tbx.tbxunit	method), 460
method), 662	(getnotes() (translate.storage.jsonl10n.118NextUnit
getlocations()	(translate.storage.tiki.TikiUnit	method), 465
method), 667	`	getnotes() (translate.storage.jsonl10n.JsonNestedUnit
getlocations()	(translate.storage.tmx.tmxunit	method), 471
method), 672		getnotes() (translate.storage.jsonl10n.JsonUnit
getlocations()	(trans-	method), 475
late.storage.trad 676	dos.TradosUnit method),	getnotes () (translate.storage.jsonl10n.WebExtensionJsonUnit method), 480
getlocations() method), 684	(translate.storage.ts2.tsunit	getnotes() (translate.storage.lisa.LISAunit method), 485
getlocations() method), 689	(translate.storage.txt.TxtUnit	getnotes() (translate.storage.mo.mounit method), 491
getlocations() method), 695	(translate.storage.utx.UtxUnit	getnotes() (translate.storage.mozilla_lang.LangUnit method), 496
getlocations()	(trans-	getnotes() (translate.storage.omegat.OmegaTUnit
=	rdfast.WordfastUnit method),	method), 503
701		getnotes() (translate.storage.php.LaravelPHPUnit
getlocations()	(translate.storage.xliff.xliffunit	method), 561
method), 708		getnotes() (translate.storage.php.phpunit method),
getnodetext() ($in n$	nodule translate.misc.ourdom),	566
398		getnotes() (translate.storage.pocommon.pounit
<pre>getNodeText()</pre>	(translate.storage.lisa.LISAunit	method), 572
method), 484	1.00 D 371.007	getnotes() (translate.storage.poxliff.PoXliffUnit
	slate.storage.poxliff.PoXliffUnit	method), 581
method), 580	(4l	getnotes() (translate.storage.properties.proppluralunit
=	(translate.storage.qph.QphUnit	method), 608
<pre>method), 636 getNodeText()</pre>	(translate.storage.tbx.tbxunit	getnotes() (translate.storage.properties.propunit method), 611
method), 661	(Hansidie.siorage.ibx.ibxumii	getnotes() (translate.storage.properties.xwikiunit
getNodeText()	(translate.storage.tmx.tmxunit	method), 619
method), 672	(ii aiisiaacisto) age.iiii.iiii.iiii.iiii	getnotes() (translate.storage.pypo.pounit method),
	slate.storage.ts2.tsunit method),	625
683	3	getnotes() (translate.storage.qm.qmunit method),
<pre>getNodeText()</pre>	(translate. storage. xliff. xliffunit	631

getnotes() (translate.storage.qph.QphUnit method), 637	<pre>getoutputname() (trans- late.convert.po2tmx.TmxOptionParser</pre>
getnotes() (translate.storage.rc.rcunit method), 643	method), 259
getnotes() (translate.storage.subtitles.SubtitleUnit	getoutputname() (trans-
method), 657	late.convert.po2wordfast.WfOptionParser
getnotes() (translate.storage.tbx.tbxunit method),	method), 264
662	getoutputname() (trans-
getnotes() (translate.storage.tiki.TikiUnit method), 667	late.filters.pofilter.FilterOptionParser method), 353
getnotes() (translate.storage.tmx.tmxunit method),	getoutputname() (trans-
672	late.misc.optrecurse.RecursiveOptionParser
getnotes() (translate.storage.trados.TradosUnit	method), 394
method), 677	getoutputname() (trans-
getnotes () (translate.storage.ts2.tsunit method), 684	late.tools.poconflicts.ConflictOptionParser
getnotes() (translate.storage.txt.TxtUnit method), 689	method), 716
	getoutputname() (trans- late.tools.pogrep.GrepOptionParser method),
getnotes() (translate.storage.utx.UtxUnit method), 695	720
getnotes() (translate.storage.wordfast.WordfastUnit	getoutputname() (trans-
method), 701	late.tools.porestructure.SplitOptionParser
getnotes() (translate.storage.xliff.xliffunit method),	method), 723
708	getoutputname() (trans-
getnumbers() (in module trans-	late.tools.poterminology.TerminologyOptionParser
late.filters.decoration), 351	method), 727 getoutputoptions() (trans-
<pre>getobject() (in module translate.storage.factory),</pre>	late.convert.convert.ArchiveConvertOptionParser
getoofile() (translate.storage.oo.oomultifile	method), 238
method), 507	getoutputoptions() (trans-
getoutput() (translate.storage.dtd.dtdunit method),	late.convert.convert.ConvertOptionParser
430	method), 242
getoutput() (translate.storage.oo.oofile method),	getoutputoptions() (trans-
506	late.convert.po2moz.MozConvertOptionParser
getoutput() (translate.storage.oo.ooline method),	method), 254
506	getoutputoptions() (trans-
getoutput() (translate.storage.oo.oounit method), 507	late.convert.po2tmx.TmxOptionParser method), 259
getoutput() (translate.storage.php.LaravelPHPUnit	getoutputoptions() (trans-
method), 561	late.convert.po2wordfast.WfOptionParser
getoutput() (translate.storage.php.phpunit method),	method), 264
566	getoutputoptions() (trans-
getoutput() (translate.storage.properties.propunit	late.filters.pofilter.FilterOptionParser method),
method), 611	353
getoutput() (translate.storage.properties.xwikiunit	getoutputoptions() (trans-
method), 619	late.misc.optrecurse.RecursiveOptionParser
getoutput () (translate.storage.rc.rcunit method), 643	method), 395
getoutputname() (trans-	getoutputoptions() (trans-
late.convert.convert.ArchiveConvertOptionParse method), 238	r late.tools.poconflicts.ConflictOptionParser method), 716
getoutputname() (trans-	getoutputoptions() (trans-
late.convert.convert.ConvertOptionParser	late.tools.pogrep.GrepOptionParser method),
method), 242	720
getoutputname() (trans-	getoutputoptions() (trans-
late.convert.po2moz.MozConvertOptionParser method), 254	late.tools.porestructure.SplitOptionParser method), 723

getoutputoptions() (trans-	
late.tools.poterminology.TerminologyOptionPars method), 727	er method), 436 getprojectstyle() (trans-
getParser() (translate.misc.ourdom.ExpatBuilderNS	late.storage.html.POHTMLParser method),
method), 397	434
getparts() (translate.storage.oo.ooline method), 506	
getpassthroughoptions() (trans-	method), 442
late.convert.convert.ArchiveConvertOptionParset method), 238	rgetprojectstyle() (translate.storage.ini.inifile method), 447
getpassthroughoptions() (trans-	getprojectstyle() (trans-
late.convert.convert.ConvertOptionParser method), 242	late.storage.jsonl10n.ARBJsonFile method), 453
getpassthroughoptions() (trans-	<pre>getprojectstyle() (trans-</pre>
late.convert.po2moz.MozConvertOptionParser method), 254	late.storage.jsonl10n.GoI18NJsonFile method), 458
getpassthroughoptions() (trans-	getprojectstyle() (trans-
late.convert.po2tmx.TmxOptionParser method), 259	late.storage.jsonl10n.I18NextFile method), 462
getpassthroughoptions() (trans-	getprojectstyle() (trans-
late.convert.po2wordfast.WfOptionParser	late.storage.jsonl10n.JsonFile method), 467
method), 264	getprojectstyle() (trans-
getpassthroughoptions() (trans-	late.storage.jsonl10n.JsonNestedFile method),
$late.filters.pofilter.FilterOptionParser\ method),$	469
353	getprojectstyle() (trans-
getpassthroughoptions() (trans-	late. storage. js on 110 n. WebExtensionJs on File
late.misc.optrecurse.RecursiveOptionParser	method), 477
method), 395	getprojectstyle() (translate.storage.lisa.LISAfile
getpassthroughoptions() (trans-	method), 482
late.tools.poconflicts.ConflictOptionParser method), 716	getprojectstyle() (translate.storage.mo.mofile method), 488
getpassthroughoptions() (trans-	getprojectstyle() (trans-
late.tools.pogrep.GrepOptionParser method), 720	late.storage.mozilla_lang.LangStore method), 494
getpassthroughoptions() (trans-	getprojectstyle() (trans-
late.tools.porestructure.SplitOptionParser method), 724	late.storage.omegat.OmegaTFile method), 499
getpassthroughoptions() (trans-	getprojectstyle() (trans-
late.tools.poterminology.TerminologyOptionPars method), 727	er late.storage.omegat.OmegaTFileTab method), 501
getpos() (translate.storage.html.htmlfile method), 436	getprojectstyle() (trans-
getpos() (translate.storage.html.POHTMLParser method), 434	late.storage.php.LaravelPHPFile method), 559
getprojectstyle() (trans- late.storage.base.DictStore method), 404	<pre>getprojectstyle() (translate.storage.php.phpfile method), 563</pre>
getprojectstyle() (trans-	getprojectstyle() (trans-
late.storage.base.TranslationStore method),	late.storage.pocommon.pofile method), 568
409	getprojectstyle() (trans-
getprojectstyle() (trans-	late.storage.poheader.poheader method),
late.storage.catkeys.CatkeysFile method),	574
416	getprojectstyle() (trans-
getprojectstyle() (trans-	late.storage.poxliff.PoXliffFile method), 577
late.storage.csvl10n.csvfile method), 421	getprojectstyle() (trans-
getprojectstyle() (translate.storage.dtd.dtdfile	late.storage.properties.gwtfile method), 596
method), 428	getprojectstyle() (trans-

late.storage.properties.javafile 598	method),	getprojectstyle() <i>method</i>), 665	(translate.storage.tiki.TikiStore
getprojectstyle() late.storage.properties.javautf16file 600	(trans- method),	getprojectstyle() method), 670	(translate.storage.tmx.tmxfile
getprojectstyle() late.storage.properties.javautf8file 602	(trans- method),	679	(trans- dos.TradosTxtTmFile method),
getprojectstyle()	(trans-	getprojectstyle() method), 681	(translate.storage.ts2.tsfile
late.storage.properties.joomlafile 604	method),	<pre>getprojectstyle() method), 687</pre>	(translate.storage.txt.TxtFile
getprojectstyle() late.storage.properties.propfile	(trans- method),	getprojectstyle() method), 692	(translate.storage.utx.UtxFile
605 getprojectstyle()	(trans-	getprojectstyle()	-trans) ,rdfast.WordfastTMFile method
late. storage. properties. strings file	method),	699	
613 getprojectstyle()	(trans-	<pre>getprojectstyle() method), 705</pre>	(translate.storage.xliff.xlifffile
late.storage.properties.stringsutf8file 615	method),	getrestype() (tran method), 581	slate.storage.poxliff.PoXliffUnit
getprojectstyle() late.storage.properties.xwikifile	(trans- method),	getrestype() method),708	(translate.storage.xliff.xliffunit
617	,,	getsourcelanguage	() (trans-
getprojectstyle()	(trans-	late.storage.bas	e.DictStore method), 404
late. storage. properties. XWiki Full Pag	e	getsourcelanguage	
<pre>method), 592 getprojectstyle()</pre>	(trans-	late.storage.bas 409	e.TranslationStore method),
late. storage. properties. XWikiPageProperties and the properties and the properties and the properties and the properties are properti	perties	getsourcelanguage	() (trans-
<pre>method), 594 getprojectstyle() (translate.storage.p</pre>	oypo.pofile	late.storage.cat 416	keys.CatkeysFile method),
method), 622		getsourcelanguage	() (trans-
<pre>getprojectstyle() (translate.storage</pre>	.qm.qmfile		l10n.csvfile method), 421 () (translate.storage.dtd.dtdfile
<pre>getprojectstyle() (translate.storage.qp</pre>	h.QphFile	<pre>method), 428 getsourcelanguage</pre>	() (trans-
getprojectstyle() (translate.stora	ge.rc.rcfile	late.storage.htm	al.htmlfile method), 436
<pre>method), 640 getprojectstyle()</pre>	(trans-	getsourcelanguage	al.POHTMLParser method),
late. storage. subtitles. Adv SubStation Advisor Adv	`	434	
<pre>method), 647 getprojectstyle()</pre>	(trans-	getsourcelanguage	() (trans- l.icalfile method), 442
late.storage.subtitles.MicroDVDFile 649	method),		() (translate.storage.ini.inifile
getprojectstyle()	(trans-	getsourcelanguage	() (trans-
late.storage.subtitles.SubRipFile 651	method),		nl10n.ARBJsonFile method),
<pre>getprojectstyle() late.storage.subtitles.SubStationAlpha method), 653</pre>	(trans- aFile	getsourcelanguage late.storage.jso method), 458	() (trans- nl10n.GoI18NJsonFile
getprojectstyle()	(trans-	getsourcelanguage	() (trans-
late.storage.subtitles.SubtitleFile 654	method),	late.storage.jso 462	nl10n.118NextFile method),
<pre>getprojectstyle() (translate.storage</pre>	.tbx.tbxfile	getsourcelanguage late.storage.jso	() (trans- nl10n.JsonFile method), 467

<pre>getsourcelanguage() late.storage.jsonl10n.JsonNestedFile 469</pre>	(trans- method),	getsourcelanguage() (tra- late.storage.properties.XWikiFullPage method), 592	ans-
<pre>getsourcelanguage() late.storage.jsonl10n.WebExtensionJso method), 477</pre>	(trans- onFile	getsourcelanguage() (tra- late.storage.properties.XWikiPageProperties method), 594	ans- s
<pre>getsourcelanguage()</pre>		getsourcelanguage() (tra	ans-
late.storage.lisa.LISAfile method), 482		late.storage.pypo.pofile method), 623	
<pre>getsourcelanguage() (translate.storage.m method), 488</pre>	no.mofile	<pre>getsourcelanguage() (translate.storage.qm.qn</pre>	ıfile
<pre>getsourcelanguage()</pre>	(trans-	getsourcelanguage() (tra	ans-
late.storage.mozilla_lang.LangStore	method),	late.storage.qph.QphFile method), 634	
494		getsourcelanguage() (translate.storage.rc.rc	cfile
getsourcelanguage()	(trans-	method), 641	
	method),	3 3	ans-
499		late.storage.subtitles.AdvSubStationAlphaFi	ile
<pre>getsourcelanguage()</pre>	(trans-	method), 647	
late.storage.omegat.OmegaTFileTab	method),		ans-
501		late.storage.subtitles.MicroDVDFile metho	od),
getsourcelanguage()	(trans-	649	
	method),		ans-
559		late.storage.subtitles.SubRipFile metho	oa),
<pre>getsourcelanguage()</pre>	(trans-	651	
late.storage.php.phpfile method), 563	(4		ans-
<pre>getsourcelanguage()</pre>	(trans-	late.storage.subtitles.SubStationAlphaFile	
late.storage.pocommon.pofile method).		method), 653	~~~
getsourcelanguage()			ans-
late.storage.poxliff.PoXliffFile method)	(trans-	late.storage.subtitles.SubtitleFile metho 654	9a),
getsourcelanguage()			vfila
<pre>late.storage.properties.gwtfile method) getsourcelanguage()</pre>	(trans-	getsourcelanguage() (translate.storage.tbx.tbx method), 659	gue
	method),		ans-
598	memoa),	late.storage.tiki.TikiStore method), 665	ırıs-
getsourcelanguage()	(trans-		ans-
	method),	late.storage.tmx.tmxfile method), 670	ıııs
600	memou),		ans-
getsourcelanguage()	(trans-	late.storage.trados.TradosTxtTmFile metho	
late.storage.properties.javautf8file	`		<i>Ju)</i> ,
602	,	getsourcelanguage() (translate.storage.ts2.ts	sfile
getsourcelanguage()	(trans-	method), 681	.,
	method),		ans-
604	,,	late.storage.txt.TxtFile method), 687	
<pre>getsourcelanguage()</pre>	(trans-		ans-
	method),	late.storage.utx.UtxFile method), 692	
606		•	ans-
<pre>getsourcelanguage()</pre>	(trans-	late.storage.wordfast.WordfastTMFile metho	od),
late.storage.properties.stringsfile	method),	699	
613		getsourcelanguage() (tra	ans-
<pre>getsourcelanguage()</pre>	(trans-	late.storage.xliff.xlifffile method), 705	
late.storage.properties.stringsutf8file	method),	getstartlength() (translate.search.match.match	cher
615		method), 401	
<pre>getsourcelanguage()</pre>	(trans-	getstartlength() (tra	ans-
late.storage.properties.xwikifile	method),	late.search.match.terminologymatcher	
617		method), 402	

getstoplength() (translate.search.matc	ch.matcher	late.storage.jsonl10n.JsonFile method	d), 467
method), 401		gettargetlanguage()	(trans
getstoplength()	(trans-	late. storage. js on 110 n. Js on Nested File	method)
late.search.match.terminologymatche	er .	469	
method), 402		<pre>gettargetlanguage()</pre>	(trans
<pre>getsubfilename() (translate.storage.oo.o</pre>	oomultifile	late.storage.jsonl10n.WebExtensionJs method), 477	sonFile
getsubfilesrc() (translate.storage.oo. method), 507	oomultifile	gettargetlanguage() late.storage.lisa.LISAfile method), 48	(trans
gettarget() (translate.storage.lise method), 485	a.LISAunit	gettargetlanguage() (translate.storage. method), 488	
gettarget() (translate.storage.poxliff.F method), 581	PoXliffUnit	gettargetlanguage()	(trans- method)
gettarget() (translate.storage.qp	h.QphUnit	494	
method), 637		gettargetlanguage()	(trans
gettarget() (translate.storage.tbx.tbxunit 662	method),	late.storage.omegat.OmegaTFile 499	method)
gettarget() (translate.storage.tmx.tmxuni 673	t method),	<pre>gettargetlanguage() late.storage.omegat.OmegaTFileTab</pre>	(trans- method)
<pre>gettarget() (translate.storage.ts2.tsunit</pre>	method),	501	
684		gettargetlanguage()	(trans
gettarget() (<i>translate.storage.xliff.xliffuni</i> 709	t method),	late.storage.php.LaravelPHPFile 559	method)
gettargetlanguage()	(trans-	<pre>gettargetlanguage()</pre>	(trans
late.storage.base.DictStore method),		late.storage.php.phpfile method), 564	
gettargetlanguage()	(trans- method),	gettargetlanguage() late.storage.pocommon.pofile method	(trans 1), 569
409		<pre>gettargetlanguage()</pre>	(trans
gettargetlanguage()	(trans- method),	late.storage.poheader.poheader 574	method)
416		gettargetlanguage()	(trans
gettargetlanguage()	(trans-	late.storage.poxliff.PoXliffFile method	
late.storage.csvl10n.csvfile method),		gettargetlanguage()	(trans
gettargetlanguage() (translate.storage.	dtd.dtdfile	late.storage.properties.gwtfile method	
method), 428		<pre>gettargetlanguage()</pre>	(trans
gettargetlanguage() late.storage.html.htmlfile method), 43		late.storage.properties.javafile 598	method)
gettargetlanguage()		<pre>gettargetlanguage()</pre>	(trans
late.storage.html.POHTMLParser 434	method),	late.storage.properties.javautf16file 600	method)
gettargetlanguage()	(trans-	<pre>gettargetlanguage()</pre>	(trans
late.storage.ical.icalfile method), 442 gettargetlanguage() (translate.storage		late.storage.properties.javautf8file 602	method)
method), 447		<pre>gettargetlanguage()</pre>	(trans
gettargetlanguage()	(trans-	late.storage.properties.joomlafile	method)
late.storage.jsonl10n.ARBJsonFile	method),	604	
453		gettargetlanguage()	(trans
gettargetlanguage()	e (trans-	late.storage.properties.propfile 606	method)
method), 458		<pre>gettargetlanguage()</pre>	(trans
gettargetlanguage()	(trans-	late.storage.properties.stringsfile	method)
late.storage.jsonl10n.I18NextFile	method),	613	7.
463	(1	<pre>gettargetlanguage()</pre>	(trans
gettargetlanguage()	(trans-	late.storage.properties.stringsutf8file	method)

615	method), 406
$\begin{tabular}{ll} gettargetlanguage() & (trans-\\ late.storage.properties.xwikifile & method), \end{tabular}$	$late. storage. base. Translation Unit \\ method),$
617	412
<pre>gettargetlanguage() (trans- late.storage.properties.XWikiFullPage method), 592</pre>	gettargetlen() (trans- late.storage.catkeys.CatkeysUnit method), 418
<pre>gettargetlanguage() (trans-</pre>	gettargetlen() (translate.storage.csvl10n.csvunit
late.storage.properties.XWikiPageProperties method), 594	method), 424 gettargetlen() (translate.storage.dtd.dtdunit
<pre>gettargetlanguage() (trans-</pre>	method), 430
late.storage.pypo.pofile method), 623	gettargetlen() (translate.storage.html.htmlunit
<pre>gettargetlanguage() (translate.storage.qm.qmfile</pre>	<pre>method), 439 gettargetlen() (translate.storage.ical.icalunit</pre>
<pre>gettargetlanguage() (trans-</pre>	method), 444
late.storage.qph.QphFile method), 634	gettargetlen() (translate.storage.ini.iniunit
<pre>gettargetlanguage() (translate.storage.rc.rcfile</pre>	
method), 641	gettargetlen() (trans-
$\begin{tabular}{l} gettargetlanguage() & (trans-\\ late.storage.subtitles.AdvSubStationAlphaFile \end{tabular}$	late.storage.jsonl10n.ARBJsonUnit method), 455
method), 647	gettargetlen() (trans-
<pre>gettargetlanguage() (trans-</pre>	late. storage. js on l10 n. Go I18 NJ son Unit
$late. storage. subtitles. {\it MicroDVDFile} method),$	method), 460
649	gettargetlen() (trans-
gettargetlanguage() (trans-	late.storage.jsonl10n.I18NextUnit method),
late.storage.subtitles.SubRipFile method),	465
651	gettargetlen() (trans-
gettargetlanguage() (trans-	late.storage.jsonl10n.JsonNestedUnit method),
late.storage.subtitles.SubStationAlphaFile	472
<pre>method), 653 gettargetlanguage() (trans-</pre>	gettargetlen() (trans- late.storage.jsonl10n.JsonUnit method),
late.storage.subtitles.SubtitleFile method), 654	late.storage.jsonl10n.JsonUnit method), 475 gettargetlen() (trans-
gettargetlanguage() (translate.storage.tbx.tbxfile	late.storage.jsonl10n.WebExtensionJsonUnit
method), 659	method), 480
	gettargetlen() (translate.storage.lisa.LISAunit method), 485
<pre>gettargetlanguage() (trans-</pre>	
late.storage.tmx.tmxfile method), 670	method), 491
<pre>gettargetlanguage() (trans-</pre>	gettargetlen() (trans-
late.storage.trados.TradosTxtTmFile method), 679	late.storage.mozilla_lang.LangUnit method), 496
<pre>gettargetlanguage() (translate.storage.ts2.tsfile</pre>	<pre>gettargetlen() (trans- late.storage.omegat.OmegaTUnit method),</pre>
<pre>gettargetlanguage() (trans-</pre>	503
late.storage.txt.TxtFile method), 687	gettargetlen() (trans-
<pre>gettargetlanguage() (trans- late.storage.utx.UtxFile method), 692</pre>	late.storage.php.LaravelPHPUnit method), 561
gettargetlanguage() (trans-	gettargetlen() (translate.storage.php.phpunit
$late. storage. word fast. Word fast TMFile\ method),$	method), 566
699	gettargetlen() (trans-
gettargetlanguage() (trans-	late.storage.pocommon.pounit method),
late.storage.xliff.xlifffile method), 706 gettargetlen() (translate.storage.base.DictUnit	572 gettargetlen() (trans-
gettargetren() (nansiaie.siorage.base.Dicionii	geceargeeren() (nans-

late.storage.poxliff.PoXliffUnit method), 581	<pre>gettemplatename() (trans- late.convert.po2wordfast.WfOptionParser</pre>
gettargetlen() (trans- late.storage.properties.proppluralunit method),	method), 264 gettemplatename() (trans-
608	late.filters.pofilter.FilterOptionParser method),
gettargetlen() (trans-	353
late.storage.properties.propunit method), 611	gettemplatename() (trans- late.misc.optrecurse.RecursiveOptionParser
gettargetlen() (trans-	method), 395
late.storage.properties.xwikiunit method),	gettemplatename() (trans-
619 gettargetlen() (translate.storage.pypo.pounit	late.tools.poconflicts.ConflictOptionParser method), 716
<pre>gettargetlen() (translate.storage.pypo.pounit method), 625</pre>	gettemplatename() (trans-
gettargetlen() (translate.storage.qm.qmunit method), 631	late.tools.pogrep.GrepOptionParser method), 720
gettargetlen() (translate.storage.qph.QphUnit	gettemplatename() (trans-
method), 637	late.tools.porestructure.SplitOptionParser
gettargetlen() (translate.storage.rc.rcunit	method), 724
method), 643	gettemplatename() (trans-
gettargetlen() (trans-	late. tools. poterminology. Terminology Option Parser
late.storage.subtitles.SubtitleUnit method),	method), 727
657	<pre>getText() (in module translate.misc.xml_helpers),</pre>
gettargetlen() (translate.storage.tbx.tbxunit	400
method), 662	gettext() (translate.storage.oo.ooline method), 506
gettargetlen() (translate.storage.tiki.TikiUnit method), 667	<pre>gettext_country() (in module translate.lang.data),</pre>
<pre>gettargetlen() (translate.storage.tmx.tmxunit</pre>	<pre>gettext_domain() (in module translate.lang.data),</pre>
gettargetlen() (trans-	<pre>gettext_lang() (in module translate.lang.data), 364</pre>
late.storage.trados.TradosUnit method), 677	gettranslatorcomments() (trans- late.storage.poxliff.PoXliffUnit method),
gettargetlen() (translate.storage.ts2.tsunit method), 684	581 getunits() (translate.storage.base.DictStore
gettargetlen() (translate.storage.txt.TxtUnit method), 689	method), 404 getunits() (translate.storage.base.DictUnit method),
gettargetlen() (translate.storage.utx.UtxUnit	406
<pre>method), 695 gettargetlen() (trans-</pre>	getunits() (translate.storage.base.TranslationStore method), 409
late.storage.wordfast.WordfastUnit method), 701	getunits() (translate.storage.base.TranslationUnit method), 412
gettargetlen() (translate.storage.xliff.xliffunit	getunits() (translate.storage.catkeys.CatkeysFile
method), 709	method), 416
gettemplatename() (trans-	getunits() (translate.storage.catkeys.CatkeysUnit
late.convert.convert.ArchiveConvertOptionParse method), 239	r method), 419 getunits() (translate.storage.csvl10n.csvfile
gettemplatename() (trans-	method), 421
late.convert.convert.ConvertOptionParser	getunits() (translate.storage.csvl10n.csvunit
method), 242	method), 424
gettemplatename() (trans-	getunits() (translate.storage.directory.Directory
late.convert.po2moz.MozConvertOptionParser	method), 426
method), 254	getunits()(translate.storage.dtd.dtdfile method), 428
gettemplatename() (trans-	<pre>getunits() (translate.storage.dtd.dtdunit method),</pre>
late.convert.po2tmx.TmxOptionParser	430
method), 259	getunits() (translate.storage.html.htmlfile method),

436	getunits() (translate.storage.php.LaravelPHPUnit
getunits() (translate.storage.html.htmlunit method), 439	<pre>method), 561 getunits() (translate.storage.php.phpfile method),</pre>
getunits() (translate.storage.html.POHTMLParser	564
<pre>method), 434 getunits() (translate.storage.ical.icalfile method),</pre>	getunits() (translate.storage.php.phpunit method), 566
442	getunits() (translate.storage.pocommon.pofile
getunits() (translate.storage.ical.icalunit method), 444	method), 569 getunits() (translate.storage.pocommon.pounit
getunits() (translate.storage.ini.inifile method), 447	method), 572
getunits() (translate.storage.ini.iniunit method), 449 getunits() (translate.storage.jsonl10n.ARBJsonFile	getunits() (translate.storage.poxliff.PoXliffFile method), 577
method), 453	getunits() (translate.storage.poxliff.PoXliffUnit
getunits() (translate.storage.jsonl10n.ARBJsonUnit	method), 581
method), 455 getunits() (translate.storage.jsonl10n.GoI18NJsonFil	getunits() (translate.storage.properties.gwtfile e method), 596
method), 458	getunits() (translate.storage.properties.javafile
getunits()(translate.storage.jsonl10n.GoI18NJsonUn	
method), 460	getunits() (translate.storage.properties.javautf16file method), 600
getunits() (translate.storage.jsonl10n.I18NextFile method), 463	getunits() (translate.storage.properties.javautf8file
getunits() (translate.storage.jsonl10n.118NextUnit	method), 602
method), 465	getunits() (translate.storage.properties.joomlafile
getunits() (translate.storage.jsonl10n.JsonFile	method), 604
method), 468 getunits() (translate.storage.jsonl10n.JsonNestedFile	getunits() (translate.storage.properties.propfile method), 606
method), 469	getunits() (translate.storage.properties.proppluralunit
,,,	
getunits()(translate.storage.jsonl10n.JsonNestedUnit	method), 608
getunits() (translate.storage.jsonl10n.JsonNestedUnit method), 472	method), 608 getunits() (translate.storage.properties.propunit
<pre>getunits() (translate.storage.jsonl10n.JsonNestedUnit</pre>	method), 608 getunits() (translate.storage.properties.propunit method), 611
getunits() (translate.storage.jsonl10n.JsonNestedUnit method), 472	method), 608 getunits() (translate.storage.properties.propunit method), 611 getunits() (translate.storage.properties.stringsfile sonFile method), 613
<pre>getunits() (translate.storage.jsonl10n.JsonNestedUnit</pre>	method), 608 getunits() (translate.storage.properties.propunit method), 611 getunits() (translate.storage.properties.stringsfile sonFile method), 613 getunits() (translate.storage.properties.stringsutf8file
getunits() (translate.storage.jsonl10n.JsonNestedUnit method), 472 getunits() (translate.storage.jsonl10n.JsonUnit method), 475 getunits() (translate.storage.jsonl10n.WebExtensionJ. method), 477 getunits() (translate.storage.jsonl10n.WebExtensionJ.	method), 608 getunits() (translate.storage.properties.propunit method), 611 getunits() (translate.storage.properties.stringsfile sonFile method), 613 getunits() (translate.storage.properties.stringsutf8file sonUnit method), 615
getunits () (translate.storage.jsonl10n.JsonNestedUnit method), 472 getunits () (translate.storage.jsonl10n.JsonUnit method), 475 getunits () (translate.storage.jsonl10n.WebExtensionJ. method), 477 getunits () (translate.storage.jsonl10n.WebExtensionJ. method), 480	method), 608 getunits() (translate.storage.properties.propunit method), 611 getunits() (translate.storage.properties.stringsfile sonFile method), 613 getunits() (translate.storage.properties.stringsutf8file sonUnit method), 615 getunits() (translate.storage.properties.xwikifile
getunits () (translate.storage.jsonl10n.JsonNestedUnit method), 472 getunits () (translate.storage.jsonl10n.JsonUnit method), 475 getunits () (translate.storage.jsonl10n.WebExtensionJ. method), 477 getunits () (translate.storage.jsonl10n.WebExtensionJ. method), 480 getunits () (translate.storage.lisa.LISAfile method), 482	method), 608 getunits() (translate.storage.properties.propunit method), 611 getunits() (translate.storage.properties.stringsfile sonFile method), 613 getunits() (translate.storage.properties.stringsutf8file sonUnit method), 615 getunits() (translate.storage.properties.xwikifile method), 617 getunits() (translate.storage.properties.XWikiFullPage
getunits () (translate.storage.jsonl10n.JsonNestedUnit method), 472 getunits () (translate.storage.jsonl10n.JsonUnit method), 475 getunits () (translate.storage.jsonl10n.WebExtensionJ. method), 477 getunits () (translate.storage.jsonl10n.WebExtensionJ. method), 480 getunits () (translate.storage.lisa.LISAfile method), 482 getunits () (translate.storage.lisa.LISAunit method),	method), 608 getunits() (translate.storage.properties.propunit method), 611 getunits() (translate.storage.properties.stringsfile sonFile method), 613 getunits() (translate.storage.properties.stringsutf8file sonUnit method), 615 getunits() (translate.storage.properties.xwikifile method), 617 getunits() (translate.storage.properties.XWikiFullPage method), 592
getunits () (translate.storage.jsonl10n.JsonNestedUnit method), 472 getunits () (translate.storage.jsonl10n.JsonUnit method), 475 getunits () (translate.storage.jsonl10n.WebExtensionJ. method), 477 getunits () (translate.storage.jsonl10n.WebExtensionJ. method), 480 getunits () (translate.storage.lisa.LISAfile method), 482 getunits () (translate.storage.lisa.LISAunit method), 485	method), 608 getunits() (translate.storage.properties.propunit method), 611 getunits() (translate.storage.properties.stringsfile sonFile method), 613 getunits() (translate.storage.properties.stringsutf8file sonUnit method), 615 getunits() (translate.storage.properties.xwikifile method), 617 getunits() (translate.storage.properties.XWikiFullPage method), 592 getunits() (translate.storage.properties.XWikiPageProperties
getunits () (translate.storage.jsonl10n.JsonNestedUnit method), 472 getunits () (translate.storage.jsonl10n.JsonUnit method), 475 getunits () (translate.storage.jsonl10n.WebExtensionJ. method), 477 getunits () (translate.storage.jsonl10n.WebExtensionJ. method), 480 getunits () (translate.storage.lisa.LISAfile method), 482 getunits () (translate.storage.lisa.LISAunit method), 485 getunits () (translate.storage.mo.mofile method), 488	method), 608 getunits() (translate.storage.properties.propunit method), 611 getunits() (translate.storage.properties.stringsfile sonFile method), 613 getunits() (translate.storage.properties.stringsutf8file sonUnit method), 615 getunits() (translate.storage.properties.xwikifile method), 617 getunits() (translate.storage.properties.XWikiFullPage method), 592 getunits() (translate.storage.properties.XWikiPageProperties method), 594
getunits () (translate.storage.jsonl10n.JsonNestedUnit method), 472 getunits () (translate.storage.jsonl10n.JsonUnit method), 475 getunits () (translate.storage.jsonl10n.WebExtensionJ. method), 477 getunits () (translate.storage.jsonl10n.WebExtensionJ. method), 480 getunits () (translate.storage.lisa.LISAfile method), 482 getunits () (translate.storage.lisa.LISAunit method), 485	method), 608 getunits() (translate.storage.properties.propunit method), 611 getunits() (translate.storage.properties.stringsfile sonFile method), 613 getunits() (translate.storage.properties.stringsutf8file sonUnit method), 615 getunits() (translate.storage.properties.xwikifile method), 617 getunits() (translate.storage.properties.XWikiFullPage method), 592 getunits() (translate.storage.properties.XWikiPageProperties
getunits () (translate.storage.jsonl10n.JsonNestedUnite method), 472 getunits () (translate.storage.jsonl10n.JsonUnite method), 475 getunits () (translate.storage.jsonl10n.WebExtensionJemethod), 477 getunits () (translate.storage.jsonl10n.WebExtensionJemethod), 480 getunits () (translate.storage.lisa.LISAfile method), 482 getunits () (translate.storage.lisa.LISAunitemethod), 485 getunits () (translate.storage.mo.mofilemethod), 488 getunits () (translate.storage.mo.mounitemethod), 491 getunits () (translate.storage.mozilla_lang.LangStore)	method), 608 getunits() (translate.storage.properties.propunit method), 611 getunits() (translate.storage.properties.stringsfile sonFile method), 613 getunits() (translate.storage.properties.stringsutf8file sonUnit method), 615 getunits() (translate.storage.properties.xwikifile method), 617 getunits() (translate.storage.properties.XWikiFullPage method), 592 getunits() (translate.storage.properties.XWikiPageProperties method), 594 getunits() (translate.storage.properties.xwikiunit method), 619 getunits() (translate.storage.pypo.pofile method),
getunits () (translate.storage.jsonl10n.JsonNestedUnite method), 472 getunits () (translate.storage.jsonl10n.JsonUnite method), 475 getunits () (translate.storage.jsonl10n.WebExtensionJ.method), 477 getunits () (translate.storage.jsonl10n.WebExtensionJ.method), 480 getunits () (translate.storage.lisa.LISAfile method), 482 getunits () (translate.storage.lisa.LISAunitemethod), 485 getunits () (translate.storage.mo.mofilemethod), 488 getunits () (translate.storage.mo.mounitemethod), 491 getunits () (translate.storage.mozilla_lang.LangStoremethod), 494 getunits () (translate.storage.mozilla_lang.LangUnitemethod), 494 getunits () (translate.storage.mozilla_lang.LangUnitemethod), 494	method), 608 getunits() (translate.storage.properties.propunit method), 611 getunits() (translate.storage.properties.stringsfile sonFile method), 613 getunits() (translate.storage.properties.stringsutf8file sonUnit method), 615 getunits() (translate.storage.properties.xwikifile method), 617 getunits() (translate.storage.properties.XWikiFullPage method), 592 getunits() (translate.storage.properties.XWikiPageProperties method), 594 getunits() (translate.storage.properties.xwikiunit method), 619 getunits() (translate.storage.pypo.pofile method), 623 getunits() (translate.storage.pypo.pounit method),
getunits () (translate.storage.jsonl10n.JsonNestedUnite method), 472 getunits () (translate.storage.jsonl10n.JsonUnite method), 475 getunits () (translate.storage.jsonl10n.WebExtensionJsmethod), 477 getunits () (translate.storage.jsonl10n.WebExtensionJsmethod), 480 getunits () (translate.storage.lisa.LISAfile method), 482 getunits () (translate.storage.lisa.LISAunitemethod), 485 getunits () (translate.storage.mo.mofilemethod), 488 getunits () (translate.storage.mo.mofilemethod), 481 getunits () (translate.storage.mozilla_lang.LangStoremethod), 494 getunits () (translate.storage.mozilla_lang.LangUnitemethod), 496	method), 608 getunits() (translate.storage.properties.propunit method), 611 getunits() (translate.storage.properties.stringsfile sonFile method), 613 getunits() (translate.storage.properties.stringsutf8file sonUnit method), 615 getunits() (translate.storage.properties.xwikifile method), 617 getunits() (translate.storage.properties.XWikiFullPage method), 592 getunits() (translate.storage.properties.XWikiPageProperties method), 594 getunits() (translate.storage.properties.xwikiunit method), 619 getunits() (translate.storage.pypo.pofile method), 623 getunits() (translate.storage.pypo.pounit method), 626
getunits () (translate.storage.jsonl10n.JsonNestedUnite method), 472 getunits () (translate.storage.jsonl10n.JsonUnite method), 475 getunits () (translate.storage.jsonl10n.WebExtensionJ.method), 477 getunits () (translate.storage.jsonl10n.WebExtensionJ.method), 480 getunits () (translate.storage.lisa.LISAfile method), 482 getunits () (translate.storage.lisa.LISAunitemethod), 485 getunits () (translate.storage.mo.mofilemethod), 488 getunits () (translate.storage.mo.mounitemethod), 491 getunits () (translate.storage.mozilla_lang.LangStoremethod), 494 getunits () (translate.storage.mozilla_lang.LangUnitemethod), 494 getunits () (translate.storage.mozilla_lang.LangUnitemethod), 494	method), 608 getunits() (translate.storage.properties.propunit method), 611 getunits() (translate.storage.properties.stringsfile sonFile method), 613 getunits() (translate.storage.properties.stringsutf8file sonUnit method), 615 getunits() (translate.storage.properties.xwikifile method), 617 getunits() (translate.storage.properties.XWikiFullPage method), 592 getunits() (translate.storage.properties.XWikiPageProperties method), 594 getunits() (translate.storage.properties.xwikiunit method), 619 getunits() (translate.storage.pypo.pofile method), 623 getunits() (translate.storage.pypo.pounit method),
getunits () (translate.storage.jsonl10n.JsonNestedUnite method), 472 getunits () (translate.storage.jsonl10n.JsonUnite method), 475 getunits () (translate.storage.jsonl10n.WebExtensionJemethod), 477 getunits () (translate.storage.jsonl10n.WebExtensionJemethod), 480 getunits () (translate.storage.lisa.LISAfile method), 482 getunits () (translate.storage.lisa.LISAmitemethod), 485 getunits () (translate.storage.mo.mofilemethod), 488 getunits () (translate.storage.mo.mounitemethod), 491 getunits () (translate.storage.mozilla_lang.LangStoremethod), 494 getunits () (translate.storage.mozilla_lang.LangUnitemethod), 496 getunits () (translate.storage.omegat.OmegaTFilemethod), 499 getunits () (translate.storage.omegat.OmegaTFileTab	method), 608 getunits() (translate.storage.properties.propunit method), 611 getunits() (translate.storage.properties.stringsfile sonFile method), 613 getunits() (translate.storage.properties.stringsutf8file sonUnit method), 615 getunits() (translate.storage.properties.xwikifile method), 617 getunits() (translate.storage.properties.XWikiFullPage method), 592 getunits() (translate.storage.properties.XWikiPageProperties method), 594 getunits() (translate.storage.properties.xwikiunit method), 619 getunits() (translate.storage.pypo.pofile method), 623 getunits() (translate.storage.pypo.pounit method), 626 getunits() (translate.storage.qm.qmfile method), 629 getunits() (translate.storage.qm.qmfile method), 632
getunits () (translate.storage.jsonl10n.JsonNestedUnite method), 472 getunits () (translate.storage.jsonl10n.JsonUnite method), 475 getunits () (translate.storage.jsonl10n.WebExtensionJemethod), 477 getunits () (translate.storage.jsonl10n.WebExtensionJemethod), 480 getunits () (translate.storage.lisa.LISAfile method), 482 getunits () (translate.storage.lisa.LISAfile method), 485 getunits () (translate.storage.mo.mofile method), 486 getunits () (translate.storage.mo.mofile method), 481 getunits () (translate.storage.mozilla_lang.LangStoremethod), 494 getunits () (translate.storage.mozilla_lang.LangUnitemethod), 496 getunits () (translate.storage.omegat.OmegaTFilemethod), 499 getunits () (translate.storage.omegat.OmegaTFileTabemethod), 501	method), 608 getunits() (translate.storage.properties.propunit method), 611 getunits() (translate.storage.properties.stringsfile sonFile method), 613 getunits() (translate.storage.properties.stringsutf8file sonUnit method), 615 getunits() (translate.storage.properties.xwikifile method), 617 getunits() (translate.storage.properties.XWikiFullPage method), 592 getunits() (translate.storage.properties.XWikiPageProperties method), 594 getunits() (translate.storage.properties.xwikiunit method), 619 getunits() (translate.storage.pypo.pofile method), 623 getunits() (translate.storage.pypo.pounit method), 626 getunits() (translate.storage.qm.qmfile method), 629 getunits() (translate.storage.qm.qmfile method), 632 getunits() (translate.storage.qph.QphFile method),
getunits () (translate.storage.jsonl10n.JsonNestedUnite method), 472 getunits () (translate.storage.jsonl10n.JsonUnite method), 475 getunits () (translate.storage.jsonl10n.WebExtensionJemethod), 477 getunits () (translate.storage.jsonl10n.WebExtensionJemethod), 480 getunits () (translate.storage.lisa.LISAfile method), 482 getunits () (translate.storage.lisa.LISAmitemethod), 485 getunits () (translate.storage.mo.mofilemethod), 488 getunits () (translate.storage.mo.mounitemethod), 491 getunits () (translate.storage.mozilla_lang.LangStoremethod), 494 getunits () (translate.storage.mozilla_lang.LangUnitemethod), 496 getunits () (translate.storage.omegat.OmegaTFilemethod), 499 getunits () (translate.storage.omegat.OmegaTFileTab	method), 608 getunits() (translate.storage.properties.propunit method), 611 getunits() (translate.storage.properties.stringsfile sonFile method), 613 getunits() (translate.storage.properties.stringsutf8file sonUnit method), 615 getunits() (translate.storage.properties.xwikifile method), 617 getunits() (translate.storage.properties.XWikiFullPage method), 592 getunits() (translate.storage.properties.XWikiPageProperties method), 594 getunits() (translate.storage.properties.xwikiunit method), 619 getunits() (translate.storage.pypo.pofile method), 623 getunits() (translate.storage.pypo.pounit method), 626 getunits() (translate.storage.qm.qmfile method), 629 getunits() (translate.storage.qm.qmfile method), 632
getunits () (translate.storage.jsonl10n.JsonNestedUnite method), 472 getunits () (translate.storage.jsonl10n.JsonUnite method), 475 getunits () (translate.storage.jsonl10n.WebExtensionJemethod), 477 getunits () (translate.storage.jsonl10n.WebExtensionJemethod), 480 getunits () (translate.storage.lisa.LISAfile method), 482 getunits () (translate.storage.lisa.LISAfile method), 485 getunits () (translate.storage.mo.mofile method), 486 getunits () (translate.storage.mo.mofile method), 481 getunits () (translate.storage.mozilla_lang.LangStoremethod), 494 getunits () (translate.storage.mozilla_lang.LangUnitemethod), 496 getunits () (translate.storage.omegat.OmegaTFilemethod), 499 getunits () (translate.storage.omegat.OmegaTFileTabmethod), 501 getunits () (translate.storage.omegat.OmegaTUnit	getunits() (translate.storage.properties.propunit method), 611 getunits() (translate.storage.properties.stringsfile sonFile method), 613 getunits() (translate.storage.properties.stringsutf8file sonUnit method), 615 getunits() (translate.storage.properties.xwikifile method), 617 getunits() (translate.storage.properties.XWikiFullPage method), 592 getunits() (translate.storage.properties.XWikiPageProperties method), 594 getunits() (translate.storage.properties.xwikiunit method), 619 getunits() (translate.storage.pypo.pofile method), 623 getunits() (translate.storage.pypo.pounit method), 626 getunits() (translate.storage.qm.qmfile method), 629 getunits() (translate.storage.qm.qmunit method), 632 getunits() (translate.storage.qm.qmunit method), 632 getunits() (translate.storage.qph.QphFile method), 635

getunits() (translate.storage.rc.rcunit method), 643	getusageman() (trans-
getunits() (translate.storage.statistics.Statistics method), 645	late.convert.convert.ConvertOptionParser method), 242
getunits() (translate.storage.subtitles.AdvSubStationA	
method), 647	late.convert.po2moz.MozConvertOptionParser
getunits() (translate.storage.subtitles.MicroDVDFile	method), 254
method), 649	getusageman() (trans-
getunits() (translate.storage.subtitles.SubRipFile	late.convert.po2tmx.TmxOptionParser
method), 651	method), 259
$\verb"getunits"()" (translate.storage.subtitles.SubStationAlpha") is the state of t$	
method), 653	late.convert.po2wordfast.WfOptionParser
getunits() (translate.storage.subtitles.SubtitleFile	method), 264
method), 654	getusageman() (trans-
getunits() (translate.storage.subtitles.SubtitleUnit	late.filters.pofilter.FilterOptionParser method),
method), 657	353
getunits () (translate.storage.tbx.tbxfile method), 659	getusageman() (trans-
getunits() (translate.storage.tbx.tbxunit method), 662	late.misc.optrecurse.RecursiveOptionParser method), 395
<pre>getunits() (translate.storage.tiki.TikiStore method),</pre>	getusageman() (trans-
665	late. tools. poconflicts. Conflict Option Parser
<pre>getunits() (translate.storage.tiki.TikiUnit method),</pre>	method), 716
667	getusageman() (trans-
getunits() (translate.storage.tmx.tmxfile method), 670	late.tools.pogrep.GrepOptionParser method), 720
<pre>getunits() (translate.storage.tmx.tmxunit method),</pre>	getusageman() (trans-
673	late. tools. por estructure. Split Option Parser
$\verb"getunits"()" (translate.storage.trados.TradosTxtTmFile")$	method), 724
method), 679	getusageman() (trans-
<pre>getunits() (translate.storage.trados.TradosUnit method), 677</pre>	late.tools.poterminology.TerminologyOptionParser method), 727
getunits() (translate.storage.ts2.tsfile method), 682	getusagestring() (trans-
getunits() (translate.storage.ts2.tsunit method), 684	late.convert.convert.ArchiveConvertOptionParser
<pre>getunits() (translate.storage.txt.TxtFile method),</pre>	method), 239
687	getusagestring() (trans-
<pre>getunits() (translate.storage.txt.TxtUnit method), 690</pre>	late.convert.convert.ConvertOptionParser method), 242
<pre>getunits() (translate.storage.utx.UtxFile method),</pre>	getusagestring() (trans-
692	late.convert.po2moz.MozConvertOptionParser
<pre>getunits() (translate.storage.utx.UtxUnit method),</pre>	method), 254
695	getusagestring() (trans-
getunits() (translate.storage.wordfast.WordfastTMFile method), 699	e late.convert.po2tmx.TmxOptionParser method), 259
<pre>getunits() (translate.storage.wordfast.WordfastUnit</pre>	getusagestring() (trans-
method), 702	late.convert.po2wordfast.WfOptionParser
<pre>getunits() (translate.storage.xliff.xlifffile method),</pre>	method), 264
706	getusagestring() (trans-
getunits() (translate.storage.xliff.xliffunit method), 709	late.filters.pofilter.FilterOptionParser method), 354
<pre>getunits() (translate.storage.zip.ZIPFile method),</pre>	getusagestring() (trans-
714	late.misc.optrecurse.RecursiveOptionParser
geturls() (in module translate.filters.decoration), 351	method), 395
getusageman() (trans-	getusagestring() (trans-
late.convert.convert.ArchiveConvertOptionParse method), 239	

getusagestring() (trans- late.tools.pogrep.GrepOptionParser method),	handle_entityref() (trans- late.storage.html.htmlfile method), 437
720 getusagestring() (trans- late.tools.porestructure.SplitOptionParser	handle_entityref() (trans- late.storage.html.POHTMLParser method), 434
method), 724	handlecsvunit() (translate.convert.csv2po.csv2po
getusagestring() (trans-	method), 245
late. tools. poterminology. Terminology Option Pars	rehas_content (trans-
<pre>method), 727 getvalue() (translate.storage.base.DictUnit method),</pre>	late.storage.placeables.strelem.StringElem attribute), 538
406	has_translatable_text (trans-
<pre>getvalue() (translate.storage.jsonl10n.ARBJsonUnit</pre>	late.storage.xml_extract.extract.Translatable attribute), 711
getvalue()(translate.storage.jsonl10n.GoI18NJsonUn	itHashProgressBar (class in trans-
method), 460	late.misc.progressbar), 398
<pre>getvalue() (translate.storage.jsonl10n.I18NextUnit</pre>	
method), 465	late.storage.pypo.pounit method), 626
<pre>getvalue() (translate.storage.jsonl10n.JsonNestedUnit</pre>	method), 406
getvalue() (translate.storage.jsonl10n.JsonUnit method), 475	hasplural() (translate.storage.base.TranslationUnit method), 412
<pre>getvalue() (translate.storage.jsonl10n.WebExtensionJs</pre>	som Usujtlural () (translate.storage.catkeys.CatkeysUnit method), 419
getvariables() (in module trans-	hasplural() (translate.storage.csvl10n.csvunit
late.filters.decoration), 351	method), 424
getXMLlang() (in module trans- late.misc.xml_helpers), 400	hasplural() (translate.storage.dtd.dtdunit method), 430
getXMLspace() (in module trans- late.misc.xml_helpers), 400	hasplural() (translate.storage.html.htmlunit method), 439
GnomeChecker (class in translate.filters.checks), 285 GoI18NJsonFile (class in trans-	hasplural() (translate.storage.ical.icalunit method),
late.storage.jsonl10n), 457	hasplural() (translate.storage.ini.iniunit method),
GoI18NJsonUnit (class in trans-	449
late.storage.jsonl10n), 459	hasplural() (trans-
GrepMatch (class in translate.tools.pogrep), 719 GrepOptionParser (class in translate.tools.pogrep),	late.storage.jsonl10n.ARBJsonUnit method), 455
719	hasplural() (trans-
gu (class in translate.lang.gu), 370	late. storage. js on 110 n. Go I18 NJ son Unit
<pre>guess_encoding() (translate.storage.html.htmlfile</pre>	method), 460
method), 437	hasplural() (translate.storage.jsonl10n.I18NextUnit
guess_encoding() (trans-	method), 465
late.storage.html.POHTMLParser method),	hasplural() (trans-
434	late.storage.jsonl10n.JsonNestedUnit method),
guess_language() (in module translate.lang.team),	472
384 qwtfile (class in translate.storage.properties), 595	hasplural() (translate.storage.jsonl10n.JsonUnit method), 475
	hasplural() (trans-
Н	late.storage.jsonl10n.WebExtensionJsonUnit
handle_charref() (translate.storage.html.htmlfile	method), 480
method), 437	hasplural() (translate.storage.lisa.LISAunit
handle_charref() (trans-	method), 485
late.storage.html.POHTMLParser method), 434	hasplural() (translate.storage.mo.mounit method), 491
	hasplural() (trans-

late.storage.mozilla_lang.LangUnit method),	tribute), 698
496	header() (translate.storage.mo.mofile method), 488
hasplural() (translate.storage.omegat.OmegaTUnit method), 503	header() (translate.storage.pocommon.pofile method), 569
hasplural() (translate.storage.php.LaravelPHPUnit method), 561	header() (translate.storage.poheader.poheader method), 574
hasplural() (translate.storage.php.phpunit method),	header() (translate.storage.poxliff.PoXliffFile
566	method), 577
hasplural() (translate.storage.pocommon.pounit method), 572	header () (translate.storage.pypo.pofile method), 623 hi (class in translate.lang.hi), 372
hasplural() (translate.storage.poxliff.PoXliffUnit method), 581	htmlentitydecode() (in module trans- late.misc.quote), 399
hasplural() (trans-	htmlentityencode() (in module trans-
late.storage.properties.proppluralunit method),	late.misc.quote), 399
608	htmlfile (class in translate.storage.html), 435
hasplural() (translate.storage.properties.propunit	htmlunit (class in translate.storage.html), 438
method), 611	hy (class in translate.lang.hy), 373
hasplural() (translate.storage.properties.xwikiunit	ny (class in transtate.tang.ny), 373
method), 620	I
hasplural() (translate.storage.pypo.pounit method),	I18NextFile (class in translate.storage.jsonl10n), 462
626	I18NextUnit (class in translate.storage.jsonl10n), 464
hasplural() (translate.storage.qm.qmunit method),	ical2po (class in translate.convert.ical2po), 246
632	icalfile (class in translate.storage.ical), 441
hasplural() (translate.storage.qph.QphUnit	icalunit (class in translate.storage.ical), 443
method), 637	ignoretests (translate.lang.common.Common
hasplural()(translate.storage.rc.rcunit method), 643	attribute), 362
hasplural() (translate.storage.subtitles.SubtitleUnit	inc2po() (in module trans-
method), 657	late.convert.mozfunny2prop), 248
hasplural() (translate.storage.tbx.tbxunit method),	inc2prop() (in module trans-
662	late.convert.mozfunny2prop), 248
hasplural() (translate.storage.tiki.TikiUnit method),	index() (translate.misc.multistring.multistring
667	method), 390
hasplural() (translate.storage.tmx.tmxunit method),	indicpunc (translate.lang.common.Common at-
673	tribute), 362
hasplural() (translate.storage.trados.TradosUnit	infer_state() (translate.storage.base.DictUnit
method), 677	method), 407
hasplural() (translate.storage.ts2.tsunit method),	infer_state() (trans-
684	late.storage.base.TranslationUnit method),
hasplural() (translate.storage.txt.TxtUnit method),	412
690	infer_state() (trans-
hasplural() (translate.storage.utx.UtxUnit method),	late.storage.catkeys.CatkeysUnit method),
695	419
hasplural() (trans-	<pre>infer_state() (translate.storage.csvl10n.csvunit</pre>
late.storage.wordfast.WordfastUnit method),	method), 424
702	<pre>infer_state() (translate.storage.dtd.dtdunit</pre>
hasplural() (translate.storage.xliff.xliffunit method),	method), 430
709	<pre>infer_state() (translate.storage.html.htmlunit</pre>
hassuggestion() (trans-	method), 439
late. filters. checks. Standard Unit Checker	<pre>infer_state() (translate.storage.ical.icalunit</pre>
method), 343	method), 444
hastypecomment() (translate.storage.pypo.pounit	<pre>infer_state() (translate.storage.ini.iniunit method),</pre>
method), 626	449
he (class in translate.lang.he), 371	infer_state() (trans-
header (translate.storage.wordfast.WordfastHeader at-	late.storage.isonl10n.ARBJsonUnit method).

455	<pre>infer_state() (translate.storage.tbx.tbxunit</pre>
infer_state() (trans-	method), 662
late.storage.jsonl10n.GoI18NJsonUnit method), 460	infer_state() (translate.storage.tiki.TikiUnit method), 667
<pre>infer_state() (trans- late.storage.jsonl10n.I18NextUnit method),</pre>	<pre>infer_state()</pre>
465	<pre>infer_state() (translate.storage.trados.TradosUnit</pre>
infer_state() (trans-	method), 677
late.storage.jsonl10n.JsonNestedUnit method), 472	<pre>infer_state() (translate.storage.ts2.tsunit method),</pre>
<pre>infer_state() (translate.storage.jsonl10n.JsonUnit method), 475</pre>	<pre>infer_state()</pre>
infer_state() (trans-	<pre>infer_state() (translate.storage.utx.UtxUnit</pre>
late.storage.jsonl10n.WebExtensionJsonUnit	method), 695
method), 480	infer_state() (trans-
infer_state() (translate.storage.lisa.LISAunit method), 485	late.storage.wordfast.WordfastUnit method), 702
<pre>infer_state() (translate.storage.mo.mounit method), 491</pre>	<pre>infer_state()</pre>
infer_state() (trans-	ini2po (class in translate.convert.ini2po), 247
late.storage.mozilla_lang.LangUnit method),	inifile (class in translate.storage.ini), 446
496	init_headers() (translate.storage.mo.mofile
infer_state() (trans-	method), 488
late.storage.omegat.OmegaTUnit method), 504	<pre>init_headers() (translate.storage.pocommon.pofile</pre>
<pre>infer_state() (trans-</pre>	<pre>init_headers() (trans-</pre>
late.storage.php.LaravelPHPUnit method), 561	late.storage.poheader.poheader method), 574
<pre>infer_state() (translate.storage.php.phpunit method), 566</pre>	<pre>init_headers() (translate.storage.poxliff.PoXliffFile</pre>
<pre>method), 566 infer_state() (translate.storage.pocommon.pounit</pre>	<pre>method), 577 init_headers() (translate.storage.pypo.pofile</pre>
<pre>method), 566 infer_state() (translate.storage.pocommon.pounit</pre>	<pre>method), 577 init_headers() (translate.storage.pypo.pofile method), 623 initbody() (translate.storage.lisa.LISAfile method),</pre>
<pre>method), 566 infer_state() (translate.storage.pocommon.pounit method), 572 infer_state() (translate.storage.poxliff.PoXliffUnit method), 581 infer_state() (translate.storage.poxliff.PoXliffUnit)</pre>	<pre>method), 577 init_headers() (translate.storage.pypo.pofile method), 623 initbody() (translate.storage.lisa.LISAfile method), 482 initbody() (translate.storage.poxliff.PoXliffFile</pre>
<pre>method), 566 infer_state() (translate.storage.pocommon.pounit</pre>	<pre>method), 577 init_headers() (translate.storage.pypo.pofile method), 623 initbody() (translate.storage.lisa.LISAfile method), 482 initbody() (translate.storage.poxliff.PoXliffFile method), 577</pre>
method), 566 infer_state() (translate.storage.pocommon.pounit method), 572 infer_state() (translate.storage.poxliff.PoXliffUnit method), 581 infer_state() (trans- late.storage.properties.proppluralunit method), 608	<pre>method), 577 init_headers() (translate.storage.pypo.pofile method), 623 initbody() (translate.storage.lisa.LISAfile method), 482 initbody() (translate.storage.poxliff.PoXliffFile method), 577 initbody() (translate.storage.qph.QphFile method),</pre>
<pre>method), 566 infer_state() (translate.storage.pocommon.pounit method), 572 infer_state() (translate.storage.poxliff.PoXliffUnit method), 581 infer_state() (translate.storage.poxliff.PoXliffUnit method), 581 infer_state() (translate.storage.properties.proppluralunit method), 608 infer_state() (translate.storage.properties.proppluralunit method),</pre>	<pre>method), 577 init_headers() (translate.storage.pypo.pofile method), 623 initbody() (translate.storage.lisa.LISAfile method), 482 initbody() (translate.storage.poxliff.PoXliffFile method), 577 initbody() (translate.storage.qph.QphFile method), 635</pre>
method), 566 infer_state() (translate.storage.pocommon.pounit method), 572 infer_state() (translate.storage.poxliff.PoXliffUnit method), 581 infer_state() (trans- late.storage.properties.proppluralunit method), 608	<pre>method), 577 init_headers() (translate.storage.pypo.pofile method), 623 initbody() (translate.storage.lisa.LISAfile method), 482 initbody() (translate.storage.poxliff.PoXliffFile method), 577 initbody() (translate.storage.qph.QphFile method),</pre>
method), 566 infer_state() (translate.storage.pocommon.pounit method), 572 infer_state() (translate.storage.poxliff.PoXliffUnit method), 581 infer_state() (translate.storage.poxliff.poxliffUnit method), 608 infer_state() (translate.storage.properties.proppluralunit method), 608 infer_state() (translate.storage.properties.propunit method),	method), 577 init_headers() (translate.storage.pypo.pofile method), 623 initbody() (translate.storage.lisa.LISAfile method), 482 initbody() (translate.storage.poxliff.PoXliffFile method), 577 initbody() (translate.storage.qph.QphFile method), 635 initbody() (translate.storage.tbx.tbxfile method), 659 initbody() (translate.storage.tmx.tmxfile method), 670
method), 566 infer_state() (translate.storage.pocommon.pounit method), 572 infer_state() (translate.storage.poxliff.PoXliffUnit method), 581 infer_state() (trans- late.storage.properties.proppluralunit method), 608 infer_state() (trans- late.storage.properties.propunit method), 611 infer_state() (trans- late.storage.properties.xwikiunit method),	method), 577 init_headers() (translate.storage.pypo.pofile method), 623 initbody() (translate.storage.lisa.LISAfile method), 482 initbody() (translate.storage.poxliff.PoXliffFile method), 577 initbody() (translate.storage.qph.QphFile method), 635 initbody() (translate.storage.tbx.tbxfile method), 659 initbody() (translate.storage.tmx.tmxfile method), 670 initbody() (translate.storage.ts2.tsfile method), 682
method), 566 infer_state() (translate.storage.pocommon.pounit method), 572 infer_state() (translate.storage.poxliff.PoXliffUnit method), 581 infer_state() (translate.storage.properties.proppluralunit method), 608 infer_state() (translate.storage.properties.proppluralunit method), 608 infer_state() (translate.storage.properties.propunit method), 611 infer_state() (translate.storage.properties.xwikiunit method), 620	method), 577 init_headers() (translate.storage.pypo.pofile method), 623 initbody() (translate.storage.lisa.LISAfile method), 482 initbody() (translate.storage.poxliff.PoXliffFile method), 577 initbody() (translate.storage.qph.QphFile method), 635 initbody() (translate.storage.tbx.tbxfile method), 659 initbody() (translate.storage.tmx.tmxfile method), 670 initbody() (translate.storage.ts2.tsfile method), 682 initbody() (translate.storage.xliff.xlifffile method),
method), 566 infer_state() (translate.storage.pocommon.pounit method), 572 infer_state() (translate.storage.poxliff.PoXliffUnit method), 581 infer_state() (trans- late.storage.properties.proppluralunit method), 608 infer_state() (trans- late.storage.properties.propunit method), 611 infer_state() (trans- late.storage.properties.xwikiunit method),	method), 577 init_headers() (translate.storage.pypo.pofile method), 623 initbody() (translate.storage.lisa.LISAfile method), 482 initbody() (translate.storage.poxliff.PoXliffFile method), 577 initbody() (translate.storage.qph.QphFile method), 635 initbody() (translate.storage.tbx.tbxfile method), 659 initbody() (translate.storage.tmx.tmxfile method), 670 initbody() (translate.storage.ts2.tsfile method), 682
method), 566 infer_state() (translate.storage.pocommon.pounit method), 572 infer_state() (translate.storage.poxliff.PoXliffUnit method), 581 infer_state() (translate.storage.propular lunit method), 608 infer_state() (translate.storage.propunit method), 611 infer_state() (translate.storage.properties.xwikiunit method), 620 infer_state() (translate.storage.pypo.pounit method), 626 infer_state() (translate.storage.qm.qmunit	method), 577 init_headers() (translate.storage.pypo.pofile method), 623 initbody() (translate.storage.lisa.LISAfile method), 482 initbody() (translate.storage.poxliff.PoXliffFile method), 577 initbody() (translate.storage.qph.QphFile method), 635 initbody() (translate.storage.tbx.tbxfile method), 659 initbody() (translate.storage.tmx.tmxfile method), 670 initbody() (translate.storage.ts2.tsfile method), 682 initbody() (translate.storage.xliff.xlifffile method), 706 inittm() (translate.search.match.matcher method), 402
method), 566 infer_state() (translate.storage.pocommon.pounit	method), 577 init_headers() (translate.storage.pypo.pofile method), 623 initbody() (translate.storage.lisa.LISAfile method), 482 initbody() (translate.storage.poxliff.PoXliffFile method), 577 initbody() (translate.storage.qph.QphFile method), 635 initbody() (translate.storage.tbx.tbxfile method), 659 initbody() (translate.storage.tmx.tmxfile method), 670 initbody() (translate.storage.ts2.tsfile method), 682 initbody() (translate.storage.xliff.xlifffile method), 706 inittm() (translate.search.match.matcher method), 402 inittm() (translate.search.match.terminologymatcher
method), 566 infer_state() (translate.storage.pocommon.pounit method), 572 infer_state() (translate.storage.poxliff.PoXliffUnit method), 581 infer_state() (translate.storage.properties.proppluralunit method), 608 infer_state() (translate.storage.properties.propunit method), 611 infer_state() (translate.storage.properties.xwikiunit method), 620 infer_state() (translate.storage.pypo.pounit method), 626 infer_state() (translate.storage.qm.qmunit method), 632 infer_state() (translate.storage.qph.QphUnit	method), 577 init_headers() (translate.storage.pypo.pofile method), 623 initbody() (translate.storage.lisa.LISAfile method), 482 initbody() (translate.storage.poxliff.PoXliffFile method), 577 initbody() (translate.storage.qph.QphFile method), 635 initbody() (translate.storage.tbx.tbxfile method), 659 initbody() (translate.storage.tmx.tmxfile method), 670 initbody() (translate.storage.ts2.tsfile method), 682 initbody() (translate.storage.xliff.xlifffile method), 706 inittm() (translate.search.match.matcher method), 402 inittm() (translate.search.match.terminologymatcher method), 402
method), 566 infer_state() (translate.storage.pocommon.pounit method), 572 infer_state() (translate.storage.poxliff.PoXliffUnit method), 581 infer_state() (translate.storage.propuluralunit method), 608 infer_state() (translate.storage.properties.propuluralunit method), 611 infer_state() (translate.storage.properties.xwikiunit method), 620 infer_state() (translate.storage.pypo.pounit method), 626 infer_state() (translate.storage.qm.qmunit method), 632 infer_state() (translate.storage.qph.QphUnit method), 637	method), 577 init_headers() (translate.storage.pypo.pofile method), 623 initbody() (translate.storage.lisa.LISAfile method), 482 initbody() (translate.storage.poxliff.PoXliffFile method), 577 initbody() (translate.storage.qph.QphFile method), 635 initbody() (translate.storage.tbx.tbxfile method), 659 initbody() (translate.storage.tmx.tmxfile method), 670 initbody() (translate.storage.ts2.tsfile method), 682 initbody() (translate.storage.xliff.xlifffile method), 706 inittm() (translate.search.match.matcher method), 402 inittm() (translate.search.match.terminologymatcher method), 402 iniunit (class in translate.storage.ini), 448
method), 566 infer_state() (translate.storage.pocommon.pounit method), 572 infer_state() (translate.storage.poxliff.PoXliffUnit method), 581 infer_state() (translate.storage.propuluralunit method), 608 infer_state() (translate.storage.propuluralunit method), 611 infer_state() (translate.storage.properties.xwikiunit method), 620 infer_state() (translate.storage.pypo.pounit method), 626 infer_state() (translate.storage.qm.qmunit method), 632 infer_state() (translate.storage.qph.QphUnit method), 637 infer_state() (translate.storage.rc.rcunit method), 637	method), 577 init_headers() (translate.storage.pypo.pofile method), 623 initbody() (translate.storage.lisa.LISAfile method), 482 initbody() (translate.storage.poxliff.PoXliffFile method), 577 initbody() (translate.storage.qph.QphFile method), 635 initbody() (translate.storage.tbx.tbxfile method), 659 initbody() (translate.storage.tmx.tmxfile method), 670 initbody() (translate.storage.ts2.tsfile method), 682 initbody() (translate.storage.xliff.xlifffile method), 706 inittm() (translate.search.match.matcher method), 402 inittm() (translate.search.match.terminologymatcher method), 402 iniunit (class in translate.storage.ini), 448 insert() (translate.storage.placeables.base.Bpt
method), 566 infer_state() (translate.storage.pocommon.pounit method), 572 infer_state() (translate.storage.poxliff.PoXliffUnit method), 581 infer_state() (translate.storage.propuluralunit method), 608 infer_state() (translate.storage.propuluralunit method), 611 infer_state() (translate.storage.properties.xwikiunit method), 620 infer_state() (translate.storage.pypo.pounit method), 626 infer_state() (translate.storage.qm.qmunit method), 632 infer_state() (translate.storage.qph.QphUnit method), 637 infer_state() (translate.storage.rc.rcunit method), 643	method), 577 init_headers() (translate.storage.pypo.pofile method), 623 initbody() (translate.storage.lisa.LISAfile method), 482 initbody() (translate.storage.poxliff.PoXliffFile method), 577 initbody() (translate.storage.qph.QphFile method), 635 initbody() (translate.storage.tbx.tbxfile method), 659 initbody() (translate.storage.tmx.tmxfile method), 670 initbody() (translate.storage.ts2.tsfile method), 682 initbody() (translate.storage.xliff.xlifffile method), 706 inittm() (translate.search.match.matcher method), 402 inittm() (translate.search.match.terminologymatcher method), 402 iniunit (class in translate.storage.ini), 448 insert() (translate.storage.placeables.base.Bpt method), 509
method), 566 infer_state() (translate.storage.pocommon.pounit method), 572 infer_state() (translate.storage.poxliff.PoXliffUnit method), 581 infer_state() (translate.storage.propuluralunit method), 608 infer_state() (translate.storage.propuluralunit method), 611 infer_state() (translate.storage.properties.xwikiunit method), 620 infer_state() (translate.storage.pypo.pounit method), 626 infer_state() (translate.storage.qm.qmunit method), 632 infer_state() (translate.storage.qph.QphUnit method), 637 infer_state() (translate.storage.rc.rcunit method), 637	method), 577 init_headers() (translate.storage.pypo.pofile method), 623 initbody() (translate.storage.lisa.LISAfile method), 482 initbody() (translate.storage.poxliff.PoXliffFile method), 577 initbody() (translate.storage.qph.QphFile method), 635 initbody() (translate.storage.tbx.tbxfile method), 659 initbody() (translate.storage.tmx.tmxfile method), 670 initbody() (translate.storage.ts2.tsfile method), 682 initbody() (translate.storage.xliff.xlifffile method), 706 inittm() (translate.search.match.matcher method), 402 inittm() (translate.search.match.terminologymatcher method), 402 iniunit (class in translate.storage.ini), 448 insert() (translate.storage.placeables.base.Bpt
method), 566 infer_state() (translate.storage.pocommon.pounit method), 572 infer_state() (translate.storage.poxliff.PoXliffUnit method), 581 infer_state() (translate.storage.propuluralunit method), 608 infer_state() (translate.storage.propuluralunit method), 611 infer_state() (translate.storage.properties.xwikiunit method), 620 infer_state() (translate.storage.pypo.pounit method), 626 infer_state() (translate.storage.qm.qmunit method), 632 infer_state() (translate.storage.qph.QphUnit method), 637 infer_state() (translate.storage.rc.rcunit method), 643 infer_state() (translate.storage.rc.rcunit method), 643 infer_state() (translate.storage.rc.rcunit method), 643 infer_state() (translate.storage.rc.rcunit method), 643	method), 577 init_headers() (translate.storage.pypo.pofile method), 623 initbody() (translate.storage.lisa.LISAfile method), 482 initbody() (translate.storage.poxliff.PoXliffFile method), 577 initbody() (translate.storage.qph.QphFile method), 635 initbody() (translate.storage.tbx.tbxfile method), 659 initbody() (translate.storage.tmx.tmxfile method), 670 initbody() (translate.storage.ts2.tsfile method), 682 initbody() (translate.storage.xliff.xlifffile method), 706 inittm() (translate.search.match.matcher method), 402 inittm() (translate.search.match.terminologymatcher method), 402 iniunit (class in translate.storage.ini), 448 insert() (translate.storage.placeables.base.Bpt method), 509 insert() (translate.storage.placeables.base.Bx

	<i>method</i>), 511	late.storage.placeables.base.Bpt	method),
insert	. 0 1	509	
	<i>method</i>), 519	nsert_between()	(trans-
insert	. 0 1	late.storage.placeables.base.Bx	method),
	method), 516	517	
insert	· · · · · · · · · · · · · · · · · · ·	nsert_between()	(trans-
	method), 514	late.storage.placeables.base.Ept	method),
insert	0 1	511	,
		nsert_between()	(trans-
insert	· · · · · · · · · · · · · · · · · · ·	late.storage.placeables.base.Ex	method),
	method), 522	519	(4
insert	· · · · · · · · · · · · · · · · · · ·	nsert_between()	(trans-
i n a a m+	method), 520	late.storage.placeables.base.G ceable 516	method),
Insert	() (translate.storage.placeables.general.AltAttrPlac		(tuans
incort	method), 524 i () (translate.storage.placeables.general.XMLEntity	nsert_between()	(trans- method),
Inserc	method), 525	514	memoa),
incart	() (translate.storage.placeables.general.XMLTagPl		(trans-
THISELC	method), 527	late.storage.placeables.base.Ph	method),
insert	() (translate.storage.placeables.interfaces.BasePlaceables.interfaces.BasePlaceables.interfaces.BasePlaceables.interfaces.BasePlaceables.interfaces.BasePlaceables.interfaces.BasePlaceables.interfaces.BasePlaceables.interfaces.BasePlaceables.interfaces.BasePlaceables.interfaces.BasePlaceables.interfaces.BasePlaceables.Ba	~ *	memou),
INDCIC		nsert_between()	(trans-
insert	() (translate.storage.placeables.interfaces.Invisible		method),
	method), 531	522	,
insert	() (translate.storage.placeables.interfaces.Masking	Plaeathlèoetween ()	(trans-
	method), 532	late.storage.placeables.base.X	method),
insert	() (translate.storage.placeables.interfaces.Replacea		
		nsert_between()	(trans-
insert	$() \ (translate. storage. placeables. interfaces. Subflow)$	Place able date. storage. place ables. general. Alt Alta and the properties of the	ttrPlaceable
	method), 536	method), 524	
insert	$() \ (translate. storage. placeables. strelem. String Elematical elematics) \\$		(trans-
	method), 538	late.storage.placeables.general.XML	.EntityPlaceable
insert	() (translate.storage.placeables.terminology.Termin		
		nsert_between()	(trans-
insert	0 1	late.storage.placeables.general.XML	TagPlaceable
	method), 542	method), 527	
insert		nsert_between()	(trans-
i n a a m+	method), 547	late.storage.placeables.interfaces.Ba	isePiaceable
Insert	() (translate.storage.placeables.xliff.Ept method), 544	method), 529	(trans-
insert	· · · · · · · · · · · · · · · · · · ·	.nsert_between()	`
THISELC	method), 549	method), 531	visible I laceable
insert		nsert_between()	(trans-
INDCIC	method), 550	late.storage.placeables.interfaces.Mo	*
insert.	() (translate.storage.placeables.xliff.It method),	method), 532	usiung raceaste
		nsert_between()	(trans-
insert		late.storage.placeables.interfaces.Re	*
	method), 555	method), 534	r
insert		nsert_between()	(trans-
	method), 553	late.storage.placeables.interfaces.Su	*
insert	() (translate.storage.placeables.xliff.UnknownXML		
		nsert_between()	(trans-
insert	() (translate.storage.placeables.xliff.X method),	late.storage.placeables.strelem.Strin	gElem
	545	method), 538	
	_between() (trans- i	nsert_between()	(trans-

late.storage.placeables.terminology.Terminology	
method), 540	isalpha() (translate.misc.multistring.multistring
insert_between() (trans-	method), 390
late.storage.placeables.xliff.Bpt method), 542	isapproved() (translate.storage.poxliff.PoXliffUnit method), 581
insert_between() (trans-	isapproved() (translate.storage.xliff.xliffunit
late.storage.placeables.xliff.Bx method),	method), 709
547	isarchive() (trans-
insert_between() (trans-	late.convert.convert.ArchiveConvertOptionParser
late.storage.placeables.xliff.Ept method),	method), 239
544	isarchive() (trans-
insert_between() (trans-	late.convert.po2tmx.TmxOptionParser
late.storage.placeables.xliff.Ex method),	method), 259
549	isarchive() (trans-
insert_between() (trans-	late.convert.po2wordfast.WfOptionParser
late.storage.placeables.xliff.G method), 550	method), 264
insert_between() (trans-	isascii() (translate.misc.multistring.multistring
late.storage.placeables.xliff.It method), 552	method), 390
insert_between() (trans-	isblank() (translate.storage.base.DictUnit method),
late.storage.placeables.xliff.Ph method),	407
555	isblank() (translate.storage.base.TranslationUnit
insert_between() (trans-	method), 412
late.storage.placeables.xliff.Sub method),	isblank() (translate.storage.catkeys.CatkeysUnit
553	method), 419
insert_between() (trans-	isblank() (translate.storage.csvl10n.csvunit method),
late.storage.placeables.xliff.UnknownXML	424
method), 557	isblank() (translate.storage.dtd.dtdunit method), 430
insert_between() (trans-	isblank() (translate.storage.html.htmlunit method),
late.storage.placeables.xliff.X method), 545	439
install() (translate.misc.ourdom.ExpatBuilderNS	isblank() (translate.storage.ical.icalunit method),
method), 397	444
int2byte() (in module translate.storage.oo), 505	isblank() (translate.storage.ini.iniunit method), 449
intuplelist() (in module translate.filters.checks),	isblank() (translate.storage.jsonl10n.ARBJsonUnit
351	method), 455
InvalidBundleError, 415	isblank() (translate.storage.jsonl10n.GoI18NJsonUnit
InvalidStateObjectError, 704	method), 460
invertedpunc (translate.lang.common.Common at-	isblank() (translate.storage.jsonl10n.118NextUnit
tribute), 362	method), 465
InvisiblePlaceable (class in trans-	isblank() (translate.storage.jsonl10n.JsonNestedUnit
late.storage.placeables.interfaces), 530	method), 472
IOSChecker (class in translate.filters.checks), 291	isblank() (translate.storage.jsonl10n.JsonUnit
is_comment_end() (in module trans-	method), 475
late.storage.properties), 597	isblank() (translate.storage.jsonl10n.WebExtensionJsonUnit
is_comment_one_line() (in module trans-	method), 480
late.storage.properties), 597	isblank() (translate.storage.lisa.LISAunit method),
is_comment_start() (in module trans-	485
late.storage.properties), 597	isblank() (translate.storage.mo.mounit method), 492
is_css_entity() (in module trans-	isblank() (translate.storage.mozilla_lang.LangUnit
late.convert.dtd2po), 245	method), 496
is_iterable_but_not_string() (in module	isblank() (translate.storage.omegat.OmegaTUnit
translate.convert.po2rc), 257	method), 504
is_line_continuation() (in module trans-	isblank() (translate.storage.php.LaravelPHPUnit
late.storage.properties), 597	method), 561
isalnum() (translate.misc.multistring.multistring	isblank() (translate.storage.php.phpunit method),
TOGETTAM () (Hamman misc. munish mig. munish mig	1001ans () (nanome.orage.pnp.pnpmm memod),

566 isempty() (translate.storage.ini.inifile method), 447 isblank() (translate.storage.jsonl10n.ARBJsonFile (translate.storage.pocommon.pounit isempty() method), 572 *method*), 453 $\verb|isempty()| (translate.storage.jsonl10n.GoI18NJsonFile|$ isblank() (translate.storage.poxliff.PoXliffUnit method), 581 method), 458 isblank() (translate.storage.properties.proppluralunit (translate.storage.jsonl10n.I18NextFile isempty() *method*), 608 *method*), 463 isblank() (translate.storage.properties.propunit isempty() (translate.storage.jsonl10n.JsonFile method), 611 method), 468 isempty() (translate.storage.jsonl10n.JsonNestedFile isblank() (translate.storage.properties.xwikiunit method), 620 method), 469 isblank() (translate.storage.pypo.pounit method), isempty() (translate.storage.jsonl10n.WebExtensionJsonFile 626 method), 477 (translate.storage.lisa.LISAfile method), isblank () (translate.storage.qm.qmunit method), 632 isempty() isblank() (translate.storage.qph.QphUnit method), 482 637 isempty() (translate.storage.mo.mofile method), 488 isblank() (translate.storage.rc.rcunit method), 643 isempty() (translate.storage.mozilla_lang.LangStore isblank() (translate.storage.subtitles.SubtitleUnit method), 494 *method*), 657 (translate.storage.omegat.OmegaTFile isempty() isblank () (translate.storage.tbx.tbxunit method), 662 method), 499 isblank() (translate.storage.tiki.TikiUnit method), isempty() (translate.storage.omegat.OmegaTFileTab method), 501 isblank() (translate.storage.tmx.tmxunit method), (translate.storage.php.LaravelPHPFile isempty() 673 method), 559 isblank() (translate.storage.trados.TradosUnit isempty() (translate.storage.php.phpfile method), 564 method), 677 isempty() (translate.storage.pocommon.pofile isblank() (translate.storage.ts2.tsunit method), 684 method), 569 isblank() (translate.storage.txt.TxtUnit method), 690 (translate.storage.poxliff.PoXliffFile isempty() isblank() (translate.storage.utx.UtxUnit method), method), 577 695 isempty() (translate.storage.properties.gwtfile isblank() (translate.storage.wordfast.WordfastUnit method), 596 method), 702 isempty() (translate.storage.properties.javafile (translate.storage.xliff.xliffunit method), isblank() method), 598 709 (translate.storage.properties.javautf16file isempty() isdecimal() (translate.misc.multistring.multistring method), 600 method), 390 (translate.storage.properties.javautf8file isempty() isdigit() (translate.misc.multistring.multistring method), 602 method), 391 (translate.storage.properties.joomlafile isempty() iseditable (translate.storage.placeables.strelem.StringElem method), 604 attribute), 538 isempty() (translate.storage.properties.propfile isempty() (translate.storage.base.DictStore method), method), 606 404 (translate.storage.properties.stringsfile isempty() (translate.storage.base.TranslationStore isempty() *method*), 613 method), 409 isempty() (translate.storage.properties.stringsutf8file isempty() (translate.storage.catkeys.CatkeysFile method), 615 method), 416 (translate.storage.properties.xwikifile isempty() isempty() (translate.storage.csvl10n.csvfile method), method), 617 isempty()(translate.storage.properties.XWikiFullPage 421 isempty() (translate.storage.dtd.dtdfile method), 428 method), 592 isempty() (translate.storage.html.htmlfile method), isempty()(translate.storage.properties.XWikiPageProperties 437 method), 594 (translate.storage.html.POHTMLParser isempty() isempty() (translate.storage.pypo.pofile method), 623 method), 434 isempty() (translate.storage.qm.qmfile method), 629 isempty() (translate.storage.ical.icalfile method), 442 isempty() (translate.storage.aph.OphFile method),

635	isexcluded() (trans-
isempty() (translate.storage.rc.rcfile method), 641	late. tools. poterminology. Terminology Option Parser
<pre>isempty() (translate.storage.subtitles.AdvSubStationAlp</pre>	haFile method), 727
method), 647	isexcluded() (translate.tools.pydiff.DirDiffer
<pre>isempty() (translate.storage.subtitles.MicroDVDFile</pre>	method), 730
method), 649	isfragile(translate.storage.placeables.strelem.StringElem
<pre>isempty() (translate.storage.subtitles.SubRipFile</pre>	attribute), 538
method), 651	isfuzzy()(translate.filters.checks.StandardUnitChecker
isempty()(translate.storage.subtitles.SubStationAlphaF	
method), 653	isfuzzy() (translate.storage.base.DictUnit method),
<pre>isempty() (translate.storage.subtitles.SubtitleFile</pre>	407
method), 654	isfuzzy() (translate.storage.base.TranslationUnit
isempty() (translate.storage.tbx.tbxfile method), 660	method), 412
<pre>isempty() (translate.storage.tiki.TikiStore method),</pre>	isfuzzy() (translate.storage.catkeys.CatkeysUnit
665	method), 419
isempty() (translate.storage.tmx.tmxfile method), 670	isfuzzy() (translate.storage.csvl10n.csvunit method),
<pre>isempty() (translate.storage.trados.TradosTxtTmFile</pre>	424
method), 679	isfuzzy() (translate.storage.dtd.dtdunit method), 430
isempty() (translate.storage.ts2.tsfile method), 682	isfuzzy() (translate.storage.html.htmlunit method),
isempty() (translate.storage.txt.TxtFile method), 687	439
isempty() (translate.storage.utx.UtxFile method), 692	isfuzzy() (translate.storage.ical.icalunit method),
isempty()(translate.storage.wordfast.WordfastTMFile	445
method), 699	isfuzzy() (translate.storage.ini.iniunit method), 450
isempty() (translate.storage.xliff.xlifffile method), 706	isfuzzy() (translate.storage.in.lntant memod), 450
isexcluded() (trans-	method), 455
	risfuzzy() (translate.storage.jsonl10n.Go118NJsonUnit
method), 239	method), 460
isexcluded() (trans-	isfuzzy() (translate.storage.jsonl10n.118NextUnit
late.convert.convert.ConvertOptionParser	method), 465
method), 242	isfuzzy() (translate.storage.jsonl10n.JsonNestedUnit
isexcluded() (trans-	method), 472
late.convert.po2moz.MozConvertOptionParser	isfuzzy() (translate.storage.jsonl10n.JsonUnit
method), 254	method), 475
isexcluded() (trans-	<pre>isfuzzy() (translate.storage.jsonl10n.WebExtensionJsonUnit</pre>
late.convert.po2tmx.TmxOptionParser	method), 480
method), 260	isfuzzy() (translate.storage.lisa.LISAunit method),
isexcluded() (trans-	485
late.convert.po 2 word fast. Wf Option Parser	isfuzzy() (translate.storage.mo.mounit method), 492
method), 264	<pre>isfuzzy() (translate.storage.mozilla_lang.LangUnit</pre>
isexcluded() (trans-	method), 497
late.filters.pofilter.FilterOptionParser method),	isfuzzy() (translate.storage.omegat.OmegaTUnit
354	method), 504
isexcluded() (trans-	isfuzzy() (translate.storage.php.LaravelPHPUnit
late.misc.optrecurse.RecursiveOptionParser	method), 561
method), 395	isfuzzy() (translate.storage.php.phpunit method),
isexcluded() (trans-	566
late.tools.poconflicts.ConflictOptionParser	isfuzzy() (translate.storage.pocommon.pounit
method), 716	method), 572
isexcluded() (trans-	isfuzzy() (translate.storage.poxliff.PoXliffUnit
late.tools.pogrep.GrepOptionParser method),	method), 581
720	isfuzzy() (translate.storage.properties.proppluralunit
isexcluded() (trans-	method), 608
late.tools.porestructure.SplitOptionParser	isfuzzy() (translate.storage.properties.propunit
method), 724	method), 611

```
isfuzzy()
                 (translate.storage.properties.xwikiunit isheader() (translate.storage.mo.mounit method),
                                                                492
         method), 620
isfuzzy()
              (translate.storage.pypo.pounit method),
                                                       isheader() (translate.storage.mozilla lang.LangUnit
        626
                                                                method), 497
isfuzzy () (translate.storage.qm.qmunit method), 632
                                                       isheader()
                                                                       (translate.storage.omegat.OmegaTUnit
isfuzzy() (translate.storage.qph.QphUnit method),
                                                                method), 504
                                                                       (translate.storage.php.LaravelPHPUnit
         637
                                                       isheader()
isfuzzy() (translate.storage.rc.rcunit method), 643
                                                                method), 561
isfuzzy()
                (translate.storage.subtitles.SubtitleUnit
                                                      isheader() (translate.storage.php.phpunit method),
        method), 657
                                                                566
isfuzzy() (translate.storage.tbx.tbxunit method), 662
                                                       isheader()
                                                                          (translate.storage.pocommon.pounit
              (translate.storage.tiki.TikiUnit method),
isfuzzy()
                                                                method), 572
        668
                                                                          (translate.storage.poxliff.PoXliffUnit
                                                       isheader()
isfuzzy()
              (translate.storage.tmx.tmxunit method),
                                                                method), 581
        673
                                                       isheader()(translate.storage.properties.proppluralunit
isfuzzy()
                   (translate.storage.trados.TradosUnit
                                                                method), 608
        method), 677
                                                       isheader()
                                                                         (translate.storage.properties.propunit
isfuzzy() (translate.storage.ts2.tsunit method), 684
                                                                method), 611
isfuzzy() (translate.storage.txt.TxtUnit method), 690
                                                      isheader()
                                                                        (translate.storage.properties.xwikiunit
isfuzzy()
              (translate.storage.utx.UtxUnit method),
                                                                method), 620
         695
                                                       isheader()
                                                                     (translate.storage.pypo.pounit method),
isfuzzy()
               (translate.storage.wordfast.WordfastUnit
                                                                626
         method), 702
                                                       isheader()
                                                                      (translate.storage.qm.qmunit method),
              (translate.storage.xliff.xliffunit method),
                                                                632
isfuzzy()
         709
                                                       isheader() (translate.storage.qph.QphUnit method),
isheader() (translate.storage.base.DictUnit method),
                                                                638
        407
                                                       isheader () (translate.storage.rc.rcunit method), 643
                (translate.storage.base.TranslationUnit
isheader()
                                                      isheader()
                                                                       (translate.storage.subtitles.SubtitleUnit
                                                                method), 657
        method), 412
                 (translate.storage.catkeys.CatkeysUnit
                                                      isheader()
                                                                      (translate.storage.tbx.tbxunit method),
isheader()
         method), 419
                                                                662
isheader()
                     (translate.storage.csvl10n.csvunit
                                                      isheader()
                                                                     (translate.storage.tiki.TikiUnit method),
        method), 424
                                                                668
               (translate.storage.dtd.dtdunit method),
                                                      isheader()
                                                                     (translate.storage.tmx.tmxunit method),
isheader()
         430
                                                                673
isheader() (translate.storage.html.htmlunit method),
                                                                          (translate.storage.trados.TradosUnit
                                                       isheader()
        439
                                                                method), 677
isheader() (translate.storage.ical.icalunit method),
                                                      isheader() (translate.storage.ts2.tsunit method), 684
         445
                                                       isheader()
                                                                      (translate.storage.txt.TxtUnit method),
isheader() (translate.storage.ini.iniunit method), 450
                                                                690
isheader() (translate.storage.jsonl10n.ARBJsonUnit
                                                                      (translate.storage.utx.UtxUnit method),
                                                      isheader()
        method), 455
                                                                695
isheader() (translate.storage.jsonl10n.GoI18NJsonUnitisheader() (translate.storage.wordfast.WordfastUnit
        method), 460
                                                                method), 702
isheader()
              (translate.storage.jsonl10n.118NextUnit isheader() (translate.storage.xliff.xliffunit method),
         method), 465
isheader() (translate.storage.jsonl10n.JsonNestedUnit isidentifier()
                                                                                                     (trans-
        method), 472
                                                                late.misc.multistring.multistring
                                                                                                   method),
isheader()
                   (translate.storage.jsonl10n.JsonUnit
         method), 475
                                                       isleaf()
                                                                        (translate.storage.placeables.base.Bpt
isheader() (translate.storage.jsonl10n.WebExtensionJsonUnit
                                                                method), 509
        method), 480
                                                       isleaf()
                                                                         (translate.storage.placeables.base.Bx
isheader() (translate.storage.lisa.LISAunit method),
                                                                method), 517
         485
                                                       isleaf()
                                                                        (translate.storage.placeables.base.Ept
```

method), 511	method), 391
isleaf() (translate.storage.placeables.base.Ex method), 519	isnumeric() (translate.misc.multistring.multistring method), 391
isleaf() (translate.storage.placeables.base.G method), 516	isobsolete() (translate.storage.base.DictUnit method), 407
isleaf() (translate.storage.placeables.base.It	isobsolete() (trans-
method), 514	late.storage.base.TranslationUnit method),
isleaf() (translate.storage.placeables.base.Ph	412
method), 512	isobsolete() (translate.storage.catkeys.CatkeysUnit
isleaf() (translate.storage.placeables.base.Sub	method), 419
method), 522	isobsolete() (translate.storage.csvl10n.csvunit
isleaf() (translate.storage.placeables.base.X	method), 424
method), 520	isobsolete() (translate.storage.dtd.dtdunit method),
isleaf() (translate.storage.placeables.general.AltAttrI	
method), 524 isleaf() (translate.storage.placeables.general.XMLEn	isobsolete() (translate.storage.html.htmlunit
method), 526	isobsolete() (translate.storage.ical.icalunit
isleaf() (translate.storage.placeables.general.XMLTa	
method), 527	isobsolete() (translate.storage.ini.iniunit method),
isleaf() (translate.storage.placeables.interfaces.Basel	
method), 529	isobsolete() (trans-
isleaf() (translate.storage.placeables.interfaces.Invisi	
method), 531	456
isleaf()(translate.storage.placeables.interfaces.Mask	ing Blaceable te () (trans-
method), 532	late.storage.jsonl10n.GoI18NJsonUnit
<pre>isleaf() (translate.storage.placeables.interfaces.Replaceables.</pre>	cementPlac uuhh od), 460
method), 534	isobsolete() (trans-
<pre>isleaf() (translate.storage.placeables.interfaces.Subfload</pre>	owPlaceabldate.storage.jsonl10n.I18NextUnit method),
method), 536	465
method), 536 isleaf() (translate.storage.placeables.strelem.StringE	465 emisobsolete() (trans-
method), 536 isleaf() (translate.storage.placeables.strelem.StringEnethod), 538	465 emisobsolete() (trans- late.storage.jsonl10n.JsonNestedUnit method),
method), 536 isleaf() (translate.storage.placeables.strelem.StringEdmethod), 538 isleaf() (translate.storage.placeables.terminology.Termin	465 emisobsolete() (trans- late.storage.jsonl10n.JsonNestedUnit method), minologyPla&Eable
method), 536 isleaf() (translate.storage.placeables.strelem.StringEnethod), 538 isleaf() (translate.storage.placeables.terminology.Termethod), 540	465 emisobsolete() (trans- late.storage.jsonl10n.JsonNestedUnit method), minologyPlaEPable isobsolete() (translate.storage.jsonl10n.JsonUnit
method), 536 isleaf() (translate.storage.placeables.strelem.StringEnethod), 538 isleaf() (translate.storage.placeables.terminology.Termethod), 540 isleaf() (translate.storage.placeables.xliff.Bpt	465 emisobsolete() (trans- late.storage.jsonl10n.JsonNestedUnit method), minologyPla&Pable isobsolete() (translate.storage.jsonl10n.JsonUnit method), 475
method), 536 isleaf() (translate.storage.placeables.strelem.StringEnethod), 538 isleaf() (translate.storage.placeables.terminology.Termethod), 540 isleaf() (translate.storage.placeables.xliff.Bpt method), 542	465 emisobsolete() (trans- late.storage.jsonl10n.JsonNestedUnit method), minologyPlateable isobsolete() (translate.storage.jsonl10n.JsonUnit method), 475 isobsolete() (translate.storage.jsonl10n.JsonUnit
method), 536 isleaf() (translate.storage.placeables.strelem.StringEnemethod), 538 isleaf() (translate.storage.placeables.terminology.Termethod), 540 isleaf() (translate.storage.placeables.xliff.Bptmethod), 542 isleaf() (translate.storage.placeables.xliff.Bx	465 emisobsolete() (trans- late.storage.jsonl10n.JsonNestedUnit method), minologyPlaKEable isobsolete() (translate.storage.jsonl10n.JsonUnit method), 475 isobsolete() (trans- late.storage.jsonl10n.WebExtensionJsonUnit
method), 536 isleaf() (translate.storage.placeables.strelem.StringEnemethod), 538 isleaf() (translate.storage.placeables.terminology.Termethod), 540 isleaf() (translate.storage.placeables.xliff.Bptmethod), 542 isleaf() (translate.storage.placeables.xliff.Bxmethod), 547	465 emisobsolete() (translate.storage.jsonl10n.JsonNestedUnit method), minologyPla&Pable isobsolete() (translate.storage.jsonl10n.JsonUnit method), 475 isobsolete() (translate.storage.jsonl10n.WebExtensionJsonUnit method), 480
method), 536 isleaf() (translate.storage.placeables.strelem.StringEnethod), 538 isleaf() (translate.storage.placeables.terminology.Termethod), 540 isleaf() (translate.storage.placeables.xliff.Bptmethod), 542 isleaf() (translate.storage.placeables.xliff.Bxmethod), 547 isleaf() (translate.storage.placeables.xliff.Eptmethod), 547	465 emisobsolete() (translate.storage.jsonl10n.JsonNestedUnit method), minologyPla&Pable isobsolete() (translate.storage.jsonl10n.JsonUnit method), 475 isobsolete() (translate.storage.jsonl10n.WebExtensionJsonUnit method), 480 isobsolete() (translate.storage.lisa.LISAunit
method), 536 isleaf() (translate.storage.placeables.strelem.StringEnethod), 538 isleaf() (translate.storage.placeables.terminology.Termethod), 540 isleaf() (translate.storage.placeables.xliff.Bptmethod), 542 isleaf() (translate.storage.placeables.xliff.Bxmethod), 547 isleaf() (translate.storage.placeables.xliff.Eptmethod), 544	465 emisobsolete() (translate.storage.jsonl10n.JsonNestedUnit method), minologyPlateable isobsolete() (translate.storage.jsonl10n.JsonUnit method), 475 isobsolete() (translate.storage.jsonl10n.WebExtensionJsonUnit method), 480 isobsolete() (translate.storage.lisa.LISAunit method), 486
method), 536 isleaf() (translate.storage.placeables.strelem.StringEnethod), 538 isleaf() (translate.storage.placeables.terminology.Termethod), 540 isleaf() (translate.storage.placeables.xliff.Bptmethod), 542 isleaf() (translate.storage.placeables.xliff.Bxmethod), 547 isleaf() (translate.storage.placeables.xliff.Eptmethod), 544 isleaf() (translate.storage.placeables.xliff.Extmethod), 544	465 emisobsolete() (translate.storage.jsonl10n.JsonNestedUnit method), minologyPla&Pable isobsolete() (translate.storage.jsonl10n.JsonUnit method), 475 isobsolete() (translate.storage.jsonl10n.WebExtensionJsonUnit method), 480 isobsolete() (translate.storage.lisa.LISAunit
method), 536 isleaf() (translate.storage.placeables.strelem.StringEnethod), 538 isleaf() (translate.storage.placeables.terminology.Termethod), 540 isleaf() (translate.storage.placeables.xliff.Bptmethod), 542 isleaf() (translate.storage.placeables.xliff.Bxmethod), 547 isleaf() (translate.storage.placeables.xliff.Eptmethod), 544 isleaf() (translate.storage.placeables.xliff.Exmethod), 549	465 emisobsolete() (translate.storage.jsonl10n.JsonNestedUnit method), minologyPlakEable isobsolete() (translate.storage.jsonl10n.JsonUnit method), 475 isobsolete() (translate.storage.jsonl10n.WebExtensionJsonUnit method), 480 isobsolete() (translate.storage.lisa.LISAunit method), 486 isobsolete() (translate.storage.mo.mounit method), 492
method), 536 isleaf() (translate.storage.placeables.strelem.StringEnethod), 538 isleaf() (translate.storage.placeables.terminology.Termethod), 540 isleaf() (translate.storage.placeables.xliff.Bptmethod), 542 isleaf() (translate.storage.placeables.xliff.Bxmethod), 547 isleaf() (translate.storage.placeables.xliff.Eptmethod), 544 isleaf() (translate.storage.placeables.xliff.Eptmethod), 544 isleaf() (translate.storage.placeables.xliff.Exmethod), 549	465 emisobsolete() (translate.storage.jsonl10n.JsonNestedUnit method), minologyPla&Pable isobsolete() (translate.storage.jsonl10n.JsonUnit
method), 536 isleaf() (translate.storage.placeables.strelem.StringEnethod), 538 isleaf() (translate.storage.placeables.terminology.Termethod), 540 isleaf() (translate.storage.placeables.xliff.Bptmethod), 542 isleaf() (translate.storage.placeables.xliff.Bxmethod), 547 isleaf() (translate.storage.placeables.xliff.Eptmethod), 544 isleaf() (translate.storage.placeables.xliff.Exmethod), 549 isleaf() (translate.storage.placeables.xliff.G	465 emisobsolete() (translate.storage.jsonl10n.JsonNestedUnit method), minologyPla&Pable isobsolete() (translate.storage.jsonl10n.JsonUnit
method), 536 isleaf() (translate.storage.placeables.strelem.StringEnethod), 538 isleaf() (translate.storage.placeables.terminology.Termethod), 540 isleaf() (translate.storage.placeables.xliff.Bptmethod), 542 isleaf() (translate.storage.placeables.xliff.Bxmethod), 547 isleaf() (translate.storage.placeables.xliff.Eptmethod), 544 isleaf() (translate.storage.placeables.xliff.Exmethod), 549 isleaf() (translate.storage.placeables.xliff.Gmethod), 550	465 emisobsolete() (translate.storage.jsonl10n.JsonNestedUnit method), minologyPla&Pable isobsolete() (translate.storage.jsonl10n.JsonUnit method), 475 isobsolete() (translate.storage.jsonl10n.WebExtensionJsonUnit method), 480 isobsolete() (translate.storage.lisa.LISAunit method), 486 isobsolete() (translate.storage.mo.mounit method), 492 isobsolete() (translate.storage.mo.mounit method), late.storage.mozilla_lang.LangUnit method),
method), 536 isleaf() (translate.storage.placeables.strelem.StringEnethod), 538 isleaf() (translate.storage.placeables.terminology.Termethod), 540 isleaf() (translate.storage.placeables.xliff.Bptmethod), 542 isleaf() (translate.storage.placeables.xliff.Bxmethod), 547 isleaf() (translate.storage.placeables.xliff.Eptmethod), 544 isleaf() (translate.storage.placeables.xliff.Exmethod), 549 isleaf() (translate.storage.placeables.xliff.Gmethod), 550 isleaf() (translate.storage.placeables.xliff.It method),	465 emisobsolete() (translate.storage.jsonl10n.JsonNestedUnit method), minologyPlateable isobsolete() (translate.storage.jsonl10n.JsonUnit method), 475 isobsolete() (translate.storage.jsonl10n.WebExtensionJsonUnit method), 480 isobsolete() (translate.storage.lisa.LISAunit method), 486 isobsolete() (translate.storage.mo.mounit method), 492 isobsolete() (translate.storage.mo.mounit method), 492 isobsolete() (translate.storage.mo.mounit method), 497
method), 536 isleaf() (translate.storage.placeables.strelem.StringEnethod), 538 isleaf() (translate.storage.placeables.terminology.Termethod), 540 isleaf() (translate.storage.placeables.xliff.Bptmethod), 542 isleaf() (translate.storage.placeables.xliff.Bxmethod), 547 isleaf() (translate.storage.placeables.xliff.Eptmethod), 544 isleaf() (translate.storage.placeables.xliff.Exmethod), 549 isleaf() (translate.storage.placeables.xliff.Gmethod), 550 isleaf() (translate.storage.placeables.xliff.Itmethod), 552 isleaf() (translate.storage.placeables.xliff.Phmethod), 555	465 emisobsolete() (translate.storage.jsonl10n.JsonNestedUnit method), minologyPlatPable isobsolete() (translate.storage.jsonl10n.JsonUnit method), 475 isobsolete() (translate.storage.jsonl10n.WebExtensionJsonUnit method), 480 isobsolete() (translate.storage.lisa.LISAunit method), 486 isobsolete() (translate.storage.mo.mounit method), 492 isobsolete() (translate.storage.mo.mounit method), 492 isobsolete() (translate.storage.mo.mounit method), 497 isobsolete() (translate.storage.mo.mounit method), 497 isobsolete() (translate.storage.mo.mounit method), 504
method), 536 isleaf() (translate.storage.placeables.strelem.StringEnethod), 538 isleaf() (translate.storage.placeables.terminology.Termethod), 540 isleaf() (translate.storage.placeables.xliff.Bptmethod), 542 isleaf() (translate.storage.placeables.xliff.Bxmethod), 547 isleaf() (translate.storage.placeables.xliff.Eptmethod), 544 isleaf() (translate.storage.placeables.xliff.Exmethod), 549 isleaf() (translate.storage.placeables.xliff.Gmethod), 550 isleaf() (translate.storage.placeables.xliff.Itmethod), 552 isleaf() (translate.storage.placeables.xliff.Phmethod), 555 isleaf() (translate.storage.placeables.xliff.Sub	465 emisobsolete() (translate.storage.jsonl10n.JsonNestedUnit method), minologyPlatPable isobsolete() (translate.storage.jsonl10n.JsonUnit method), 475 isobsolete() (translate.storage.jsonl10n.WebExtensionJsonUnit method), 480 isobsolete() (translate.storage.lisa.LISAunit method), 486 isobsolete() (translate.storage.mo.mounit method), 492 isobsolete() (translate.storage.mo.mounit method), 492 isobsolete() (translate.storage.mo.mounit method), 497 isobsolete() (translate.storage.mo.mounit method), 504 isobsolete() (translate.storage.omegat.OmegaTUnit method), 504 isobsolete() (translate.storage.omegat.OmegaTUnit method),
method), 536 isleaf() (translate.storage.placeables.strelem.StringEnethod), 538 isleaf() (translate.storage.placeables.terminology.Termethod), 540 isleaf() (translate.storage.placeables.xliff.Bptmethod), 542 isleaf() (translate.storage.placeables.xliff.Bxmethod), 547 isleaf() (translate.storage.placeables.xliff.Eptmethod), 544 isleaf() (translate.storage.placeables.xliff.Exmethod), 549 isleaf() (translate.storage.placeables.xliff.Gmethod), 550 isleaf() (translate.storage.placeables.xliff.Itmethod), 552 isleaf() (translate.storage.placeables.xliff.Phmethod), 555 isleaf() (translate.storage.placeables.xliff.Submethod), 553	465 emisobsolete() (translate.storage.jsonl10n.JsonNestedUnit method), minologyPlaceable isobsolete() (translate.storage.jsonl10n.JsonUnit
method), 536 isleaf() (translate.storage.placeables.strelem.StringEnethod), 538 isleaf() (translate.storage.placeables.terminology.Termethod), 540 isleaf() (translate.storage.placeables.xliff.Bptmethod), 542 isleaf() (translate.storage.placeables.xliff.Bxmethod), 547 isleaf() (translate.storage.placeables.xliff.Eptmethod), 544 isleaf() (translate.storage.placeables.xliff.Exmethod), 549 isleaf() (translate.storage.placeables.xliff.Gmethod), 550 isleaf() (translate.storage.placeables.xliff.Itmethod), 552 isleaf() (translate.storage.placeables.xliff.Phmethod), 555 isleaf() (translate.storage.placeables.xliff.Submethod), 553 isleaf() (translate.storage.placeables.xliff.UnknownX.	### 465 ### sobsolete() (translate.storage.jsonl10n.JsonNestedUnit method), ### isobsolete() (translate.storage.jsonl10n.JsonUnit
method), 536 isleaf () (translate.storage.placeables.strelem.StringEnethod), 538 isleaf () (translate.storage.placeables.terminology.Termethod), 540 isleaf () (translate.storage.placeables.xliff.Bptmethod), 542 isleaf () (translate.storage.placeables.xliff.Bxmethod), 547 isleaf () (translate.storage.placeables.xliff.Eptmethod), 544 isleaf () (translate.storage.placeables.xliff.Exmethod), 549 isleaf () (translate.storage.placeables.xliff.Gmethod), 550 isleaf () (translate.storage.placeables.xliff.It method), 552 isleaf () (translate.storage.placeables.xliff.Phmethod), 555 isleaf () (translate.storage.placeables.xliff.Submethod), 553 isleaf () (translate.storage.placeables.xliff.UnknownXnethod), 557	### 465 ### sobsolete() (translate.storage.jsonl10n.JsonNestedUnit method), #### isobsolete() (translate.storage.jsonl10n.JsonUnit
method), 536 isleaf() (translate.storage.placeables.strelem.StringEnethod), 538 isleaf() (translate.storage.placeables.terminology.Termethod), 540 isleaf() (translate.storage.placeables.xliff.Bptmethod), 542 isleaf() (translate.storage.placeables.xliff.Bxmethod), 547 isleaf() (translate.storage.placeables.xliff.Eptmethod), 544 isleaf() (translate.storage.placeables.xliff.Exmethod), 549 isleaf() (translate.storage.placeables.xliff.Gmethod), 550 isleaf() (translate.storage.placeables.xliff.Itmethod), 552 isleaf() (translate.storage.placeables.xliff.Phmethod), 555 isleaf() (translate.storage.placeables.xliff.Submethod), 553 isleaf() (translate.storage.placeables.xliff.UnknownXmethod), 557 isleaf() (translate.storage.placeables.xliff.Xmethod), 557 isleaf() (translate.storage.placeables.xliff.Xmethod),	### 465 ### sobsolete() (translate.storage.jsonl10n.JsonNestedUnit method), ### misobsolete() (translate.storage.jsonl10n.JsonUnit
method), 536 isleaf () (translate.storage.placeables.strelem.StringEnethod), 538 isleaf () (translate.storage.placeables.terminology.Termethod), 540 isleaf () (translate.storage.placeables.xliff.Bptmethod), 542 isleaf () (translate.storage.placeables.xliff.Bxmethod), 547 isleaf () (translate.storage.placeables.xliff.Eptmethod), 544 isleaf () (translate.storage.placeables.xliff.Exmethod), 549 isleaf () (translate.storage.placeables.xliff.Gmethod), 550 isleaf () (translate.storage.placeables.xliff.It method), 552 isleaf () (translate.storage.placeables.xliff.Phmethod), 555 isleaf () (translate.storage.placeables.xliff.Submethod), 553 isleaf () (translate.storage.placeables.xliff.UnknownXnethod), 557	### 465 ### sobsolete() (translate.storage.jsonl10n.JsonNestedUnit method), #### isobsolete() (translate.storage.jsonl10n.JsonUnit

<pre>isobsolete() (translate.storage.poxliff.PoXliffUnit</pre>	
method), 581	late.convert.po2tmx.TmxOptionParser
isobsolete() (trans-	method), 260
late.storage.properties.proppluralunit method),	isrecursive() (trans-
608	late.convert.po2wordfast.WfOptionParser
isobsolete() (translate.storage.properties.propunit	method), 264
method), 611	isrecursive() (trans-
isobsolete() (translate.storage.properties.xwikiunit method), 620	late.filters.pofilter.FilterOptionParser method), 354
isobsolete() (translate.storage.pypo.pounit	isrecursive() (trans-
method), 626	late.misc.optrecurse.RecursiveOptionParser
<pre>isobsolete() (translate.storage.qm.qmunit method),</pre>	method), 395
632	isrecursive() (trans-
isobsolete() (translate.storage.qph.QphUnit method), 638	late.tools.poconflicts.ConflictOptionParser method), 716
<pre>isobsolete() (translate.storage.rc.rcunit method),</pre>	isrecursive() (trans-
643	late.tools.pogrep.GrepOptionParser method),
isobsolete() (trans-	720
late.storage.subtitles.SubtitleUnit method),	isrecursive() (trans-
657	late. tools. por estructure. Split Option Parser
<pre>isobsolete() (translate.storage.tbx.tbxunit method),</pre>	method), 724
662	isrecursive() (trans-
isobsolete() (translate.storage.tiki.TikiUnit	late. tools. poterminology. Terminology Option Parser
method), 668	method), 727
isobsolete() (translate.storage.tmx.tmxunit	$\verb isreview() \textit{(translate.filters.checks.StandardUnitChecker} $
method), 673	method), 343
<pre>isobsolete() (translate.storage.trados.TradosUnit</pre>	<pre>isreview() (translate.storage.base.DictUnit method),</pre>
method), 677	407
isobsolete() (translate.storage.ts2.tsunit method), 685	isreview() (translate.storage.base.TranslationUnit method), 412
isobsolete() (translate.storage.txt.TxtUnit method), 690	isreview() (translate.storage.catkeys.CatkeysUnit method), 419
isobsolete() (translate.storage.utx.UtxUnit	isreview() (translate.storage.csvl10n.csvunit
method), 695	method), 424
isobsolete() (trans-	isreview() (translate.storage.dtd.dtdunit method),
late.storage.wordfast.WordfastUnit method),	430
702	<pre>isreview() (translate.storage.html.htmlunit method),</pre>
isobsolete() (translate.storage.xliff.xliffunit	439
method), 709	<pre>isreview() (translate.storage.ical.icalunit method),</pre>
isocode() (in module translate.lang.poedit), 380	445
isprintable() (trans-	isreview() (translate.storage.ini.iniunit method), 450
late.misc.multistring.multistring method),	isreview() (translate.storage.jsonl10n.ARBJsonUnit
391	method), 456
ispurepunctuation() (in module trans- late.filters.decoration), 351	isreview() (translate.storage.jsonl10n.GoI18NJsonUnit method), 461
isrecursive() (trans-	isreview() (translate.storage.jsonl10n.I18NextUnit
late.convert.convert.ArchiveConvertOptionParset	
method), 239	isreview() (translate.storage.jsonl10n.JsonNestedUnit
isrecursive() (trans-	method), 472
late.convert.convert.ConvertOptionParser	isreview() (translate.storage.jsonl10n.JsonUnit
method), 242	method), 475
isrecursive() (trans-	isreview() (translate.storage.jsonl10n.WebExtensionJsonUnit
late.convert.po2moz.MozConvertOptionParser	method), 480
method), 254	<pre>isreview() (translate.storage.lisa.LISAunit method),</pre>

486		<pre>istranslatable()</pre>	(translate. storage. base. Dict Unit
<pre>isreview()</pre>	(translate.storage.mo.mounit method),	method), 407	
492		<pre>istranslatable()</pre>	(trans-
isreview() method	(translate.storage.mozilla_lang.LangUnit od), 497	late.storage.ba 412	use.TranslationUnit method),
<pre>isreview()</pre>	(translate.storage.omegat.OmegaTUnit	<pre>istranslatable()</pre>	(trans-
metho	od), 504	late.storage.ca	tkeys.CatkeysUnit method),
<pre>isreview()</pre>	(translate.storage.php.LaravelPHPUnit	419	
metho	od), 561	<pre>istranslatable()</pre>	(trans-
<pre>isreview()</pre>	(translate.storage.php.phpunit method),	late.storage.cs	vl10n.csvunit method), 424
566		<pre>istranslatable()</pre>	(translate.storage.dtd.dtdunit
<pre>isreview()</pre>	(translate.storage.pocommon.pounit	method), 430	
meth	od), 572	<pre>istranslatable()</pre>	(translate.storage.html.htmlunit
<pre>isreview()</pre>	(translate. storage. pox liff. PoX liff Unit	method), 439	
meth	od), 581	<pre>istranslatable()</pre>	(translate.storage.ical.icalunit
<pre>isreview()</pre>	(translate. storage. properties. propplural unit	method), 445	
meth	od), 608	<pre>istranslatable()</pre>	(translate.storage.ini.iniunit
<pre>isreview()</pre>	(translate.storage.properties.propunit	method), 450	
meth	od), 611	<pre>istranslatable()</pre>	(trans-
isreview() method	(translate.storage.properties.xwikiunit od), 620	late.storage.js 456	onl10n.ARBJsonUnit method),
<pre>isreview()</pre>	(translate.storage.pypo.pounit method),	<pre>istranslatable()</pre>	(trans-
626		late.storage.js	onl10n.GoI18NJsonUnit
<pre>isreview()</pre>	(translate.storage.qm.qmunit method),	method), 461	
632	, , , , , , , , , , , , , , , , , , ,	<pre>istranslatable()</pre>	(trans-
isreview()	(translate.storage.qph.QphUnit method),		onl10n.118NextUnit method),
638	· · · · · · · · · · · · · · · · · · ·	466	
<pre>isreview()</pre>	(translate.storage.rc.rcunit method), 643	<pre>istranslatable()</pre>	(trans-
<pre>isreview()</pre>	(translate.storage.subtitles.SubtitleUnit	late.storage.js	onl10n.JsonNestedUnit method),
meth	od), 657	472	
<pre>isreview()</pre>	(translate.storage.tbx.tbxunit method),	<pre>istranslatable()</pre>	(trans-
662	-	late.storage.js	onl10n.JsonUnit method),
<pre>isreview()</pre>	(translate.storage.tiki.TikiUnit method),	475	
668		<pre>istranslatable()</pre>	(trans-
isreview() 673	(translate.storage.tmx.tmxunit method),	late.storage.jso method), 480	onl10n.WebExtensionJsonUnit
<pre>isreview()</pre>	(translate.storage.trados.TradosUnit	istranslatable()	(translate.storage.lisa.LISAunit
	od), 677	method), 486	`
	(translate.storage.ts2.tsunit method), 685	istranslatable()	(translate.storage.mo.mounit
<pre>isreview()</pre>	(translate.storage.txt.TxtUnit method),	method), 492	· ·
690	,	istranslatable()	(trans-
isreview() 695	(translate.storage.utx.UtxUnit method),		ozilla_lang.LangUnit method),
isreview()	(translate.storage.wordfast.WordfastUnit	<pre>istranslatable()</pre>	(trans-
	od), 702		negat.OmegaTUnit method),
isreview()	(translate.storage.xliff.xliffunit method),	504	megan emegan emi memea),
709	(istranslatable()	(trans-
isspace()	(translate.misc.multistring.multistring		np.LaravelPHPUnit method),
	od), 391	561	p. La. area III one memou),
istitle()	(translate.misc.multistring.multistring	istranslatable()	(translate.storage.php.phpunit
	od), 391	method), 566	(ansiane.siorage.pnp.pnpunn
istranslat		istranslatable()	(trans-
	torage.placeables.strelem.StringElem		ocommon.pounit method),
	oute), 538	572	memou),

istranslatable()		(trans-	method), 430		
late.storage.po. 582	xliff.PoXliffUnit	method),	<pre>istranslated() method), 440</pre>	(translate.storage.htm	nl.htmlunit
istranslatable()		(trans-	istranslated()	(translate.storage.io	cal.icalunit
late.storage.pro	operties.proppluralun	it method),	method), 445		
608			istranslated()	(translate.storage	e.ini.iniunit
istranslatable()		(trans-	method), 450		
late.storage.pro	operties.propunit	method),	istranslated()		(trans-
611			late.storage.jsc	onl10n.ARBJsonUnit	method),
istranslatable()		(trans-	456		
late.storage.pro	operties.xwikiunit	method),	istranslated()		(trans-
620	•		late.storage.jsc	onl10n.GoI18NJsonUn	ıit
istranslatable()	(translate.storage.p	ypo.pounit	method), 461		
method), 626			istranslated()		(trans-
istranslatable()	(translate.storage	.qm.qmunit		onl10n.I18NextUnit	method),
method), 632		1 1	466		,
istranslatable()	(translate.storage.qp	oh.QphUnit	istranslated()		(trans-
method), 638	, O II	~1		onl10n.JsonNestedUni	t method),
istranslatable()	(translate.stora	ge.rc.rcunit	472		,,
method), 643	(istranslated()		(trans-
istranslatable()		(trans-		onl10n.JsonUnit	method),
	btitles.SubtitleUnit	method),	475		,,
657		,,,	istranslated()		(trans-
istranslatable()	(translate.storage	.tbx.tbxunit		onl10n.WebExtension J	`
method), 662	(method), 480		
istranslatable()	(translate.storage.t	iki.TikiUnit	istranslated()	(translate.storage.lis	a.LISAunit
method), 668	(ir aristare is reversed		method), 486	(1.01151011015101015	
istranslatable()	(translate.storage.t	mx.tmxunit	istranslated()	(translate.storage.	mo.mounit
method), 673	(*		method), 492	(*	
istranslatable()		(trans-	istranslated()		(trans-
	dos.TradosUnit	method),		ozilla_lang.LangUnit	method),
677		,	497	2	,,
istranslatable()	(translate.storag	e.ts2.tsunit	istranslated()		(trans-
method), 685	(i. diribitational dis	,0.1.52.1.511.111		negat.OmegaTUnit	method),
istranslatable()	(translate.storage	txt TxtI/nit	504		,,
method), 690	(1.61.151611615161616		istranslated()		(trans-
istranslatable()	(translate.storage.i	utx I]txI]nit		p.LaravelPHPUnit	method),
<i>method</i>), 696	(iransiane.siorage.i		562	p.Euraveil III Omi	memou),
istranslatable()		(trans-	istranslated()	(translate.storage.p	hn nhnunit
	ordfast.WordfastUnit	method),	method), 566	(transtate.storage.p	np.pnpunu
702	rajasi. Worajasi e illi	memou),	istranslated()		(trans-
istranslatable()	(translate.storage.x	liff xliffunit	• • • • • • • • • • • • • • • • • • • •	common.pounit	method),
<i>method</i>), 709	(transtate.storage.x	iijj.xiijjuriii	572	common.pounti	memou),
	(translate.storage.ba	se DictUnit	istranslated()		(trans-
<i>method</i>), 407	(iransiare.siorage.oa)	sc.Dierenni		xliff.PoXliffUnit	method),
istranslated()		(trans-	582	anggar ozang o un	memou),
	se.TranslationUnit	method),	istranslated()		(trans-
412		memou),		operties.proppluraluni	
istranslated()		(trans-	608	оролиозгрторринации	,
	tkeys.CatkeysUnit	method),	istranslated()		(trans-
419	, b. Carne y b 0 1111	memou),		operties.propunit	method),
	translate.storage.csvl	10n.csvunit	611	oponiosipropunii	cou),
<i>method</i>), 424			istranslated()		(trans-
istranslated()	(translate.storage.	dtd.dtdunit		operties.xwikiunit	method),
(/	(1	

620 istranslated() (translate.storage.pypo.pounit	isvalidinputname() late.misc.optrecurse.RecursiveOptionPar	(trans-
method), 626	method), 395	~
istranslated() (translate.storage.qm.qmunit		(trans-
method), 632	late.tools.poconflicts.ConflictOptionPars	er
istranslated() (translate.storage.qph.QphUnit	method), 717	
method), 638		(trans-
istranslated() (translate.storage.rc.rcunit method), 643	late.tools.pogrep.GrepOptionParser m 720	ethod),
istranslated() (trans-	±	(trans-
late.storage.subtitles.SubtitleUnit method), 657	late.tools.porestructure.SplitOptionParse method), 724	er
istranslated() (translate.storage.tbx.tbxunit	isvalidinputname()	(trans-
method), 663	late. tools. poterminology. Terminology Op	tionParser
istranslated() (translate.storage.tiki.TikiUnit	method), 727	
method), 668	isvisible (translate.storage.placeables.strelem	.StringElem
istranslated() (translate.storage.tmx.tmxunit	attribute), 538	2
method), 673	It (class in translate.storage.placeables.base), 51	
istranslated() (trans-	It (class in translate.storage.placeables.xliff), 55	
late.storage.trados.TradosUnit method), 677	it2po() (in module translate.convert.mozfunny.	zprop),
istranslated() (translate.storage.ts2.tsunit		trans-
method), 685	late.convert.mozfunny2prop), 248	irans-
	items() (translate.misc.dictutils.cidict method),	389
method), 690	items() (translate.storage.oo.unormali	
istranslated() (translate.storage.utx.UtxUnit	method), 507	
method), 696		(trans-
istranslated() (trans-	late.storage.placeables.base.Bpt m	ethod),
late.storage.wordfast.WordfastUnit method),	509	
702	— <u> </u>	(trans-
istranslated() (translate.storage.xliff.xliffunit method), 709	517	ethod),
isupper() (translate.misc.multistring.multistring		(trans-
method), 391		ethod),
isvalidaccelerator() (in module trans-	511	(4
late.filters.decoration), 351	_	(trans-
<pre>isvalidinputname() (trans- late.convert.convert.ArchiveConvertOptionParser</pre>		ethod),
method), 239		(trans-
isvalidinputname() (trans-		ethod),
late.convert.convert.ConvertOptionParser	516	······································
method), 242		(trans-
isvalidinputname() (trans-		ethod),
late.convert.po2moz.MozConvertOptionParser	514	
method), 254	<pre>iter_depth_first()</pre>	(trans-
isvalidinputname() (trans-	late.storage.placeables.base.Ph m	ethod),
late.convert.po2tmx.TmxOptionParser	513	
method), 260	-	(trans-
isvalidinputname() (trans-	~ ·	ethod),
late.convert.po2wordfast.WfOptionParser	522	(1
method), 264	_	(trans-
isvalidinputname() (trans-	late.storage.placeables.base.X m 521	ethod),
late.filters.pofilter.FilterOptionParser method), 354		(trans-
JJT	rcer_aehcu_rrrsc()	(mans-

late.storage.placeables.general.Alt. method), 524	AttrPlaceable	<pre>iter_depth_first() late.storage.placeables.xliff.X methor</pre>	(trans- od), 546
iter_depth_first()	(trans-		
late.storage.placeables.general.XM	ILEntityPlaced	able	
method), 526	•	ja (class in translate.lang.ja), 373	
iter_depth_first()	(trans-	java_utf8_properties_encode()	(in module
late.storage.placeables.general.XM		translate.misc.quote), 399	(in mounic
method), 527	O	javafile (class in translate.storage.proper	ties) 598
<pre>iter_depth_first()</pre>	(trans-	javapropertiesencode() (in mod	
late.storage.placeables.interfaces.l	*		ute trans-
method), 529	susci incentie	······································	*****
iter_depth_first()	(trans-	javautf16file (class in	trans-
late storage placeables interfaces l		late.storage.properties), 599	
method), 531		ablevautf8file (class in translate.storage.	
<pre>iter_depth_first()</pre>	(trans-	join() (translate.misc.multistring.multistri	ng method),
late.storage.placeables.interfaces.l	MaskingPlaced	able 391	
method), 533		joomlafile (class in translate.storage.	properties),
iter_depth_first()	(trans-	603	
late.storage.placeables.interfaces.l	ReplacementPl	laçgehlgpo (class in translate.convert.json2po), 247
method), 534		JsonFile (class in translate.storage.jsonl1)	(2n), 467
iter_depth_first()	(trans-	JsonNestedFile (class in	trans-
late.storage.placeables.interfaces.S	SubflowPlacea	ble late.storage.jsonl10n), 469	
method), 536		JsonNestedUnit (class in	trans-
iter_depth_first()	(trans-	late.storage.jsonl10n), 470	
late.storage.placeables.strelem.Stre method), 538	ingElem	JsonUnit (class in translate.storage.jsonl1)	0n), 473
iter depth first()	(trans-	K	
late.storage.placeables.terminolog	v.Terminology	Placeable Kaechecker (class in translate.filters.check	206
method), 540	,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,	Käechecker (class in translate.filters.check	(s), 296
iter_depth_first()	(trans-	kdecomments()	(trans-
late.storage.placeables.xliff.Bpt	method),	late.filters.checks.CCLicenseChecke	er method),
542	memoa),	275	
	(trans	kdecomments()	(trans-
iter_depth_first()	(trans-	late.filters.checks.DrupalChecker	method),
late.storage.placeables.xliff.Bx	method),	281	
547	(,	kdecomments()	(trans-
<pre>iter_depth_first()</pre>	(trans-	late. filters. checks. Gnome Checker	method),
late.storage.placeables.xliff.Ept	method),	287	
544	,	kdecomments() (translate.filters.checks	IOSChecker
<pre>iter_depth_first()</pre>	(trans-	method), 293	
late.storage.placeables.xliff.Ex	method),	kdecomments() (translate.filters.checks.	KdeChecker
549		method), 298	
iter_depth_first()	(trans-	kdecomments() (translate.filters.checks.L	20nChecker
late.storage.placeables.xliff.G meth	nod), 550	method), 304	
<pre>iter_depth_first()</pre>	(trans-	kdecomments()	(trans-
late.storage.placeables.xliff.It meth	nod), 552	late.filters.checks.LibreOfficeCheck	`
<pre>iter_depth_first()</pre>	(trans-	310	i memou),
late.storage.placeables.xliff.Ph	method),	kdecomments()	(trans-
555		late.filters.checks.MinimalChecker	method),
iter_depth_first()	(trans-	316	тетои),
late.storage.placeables.xliff.Sub	method),		(tuana
554	,,	kdecomments()	(trans-
iter_depth_first()	(trans-	late.filters.checks.MozillaChecker	method),
late.storage.placeables.xliff.Unkno		321	<i>(</i> :
method), 557		kdecomments()	(trans-
111011001), 551		late.filters.checks.OpenOfficeCheck	er method),

327		L	
kdecomments()	(trans-	L20nChecker (class in transla	te.filters.checks), 302
late.filters.checks.ReducedChecker 333	method),	labelsuffixes (in module 432	*
kdecomments()	(trans-	labelsuffixes (in	module trans-
late.filters.checks.StandardChecker 339	method),	late.storage.properties	, 605
kdecomments() (translate.filters.checks.Ter method), 346	mChecker	lang2po (class in translate.com lang_codes (in module transla	ate.lang.poedit), 380
key_strip() (translate.storage.properti class method), 586	es.Dialect	lang_names (in module transle LANG_TEAM_CONTACT_SNIP	
key_strip()	(trans-	late.lang.team), 383	
late. storage. properties. Dialect Flex	class	LangStore (class in translated) 493	e.storage.mozilla_lang),
method), 587		LanguageError,669	
key_strip()	(trans-	languagematch() (in mode	ıle translate.lang.data),
late.storage.properties.DialectGaia method), 587	class	364 languages (in module translat	re lang data) 364
key_strip()	(trans-	Langunit (class in translate.ste	_
late.storage.properties.DialectGwt method), 588	class	LaravelPHPFile (class in tra	enslate.storage.php), 558
key_strip()	(trans-	LaravelPHPUnit (class in tra	
late.storage.properties.DialectJava	class	lastChild (translate.misc.o. tribute), 397	ourdom.Document at-
method), 588	(4	lastChild (translate.misc.out	dom.Element attribute),
key_strip()	(trans-	397	
late.storage.properties.DialectJavaUt class method), 589	710	launch_server() (in modi 400	ule translate.misc.wsgi),
key_strip()	(trans-	<pre>length_difference() (tr</pre>	anslate.lang.af.af class
late.storage.properties.DialectJavaUt	f8 class	<i>method</i>), 357	
method), 589	,	length_difference() (tra	nslate.lang.am.am class
key_strip()	(trans-	method), 358	
late.storage.properties.DialectJoomla method), 589	ı class	<pre>length_difference() (tr method), 359</pre>	anslate.lang.ar.ar class
key_strip()	(trans-	length_difference() (tra	anslate.lang.bn.bn class
late.storage.properties.DialectMozillo method), 590	a class	<pre>method), 359 length_difference()</pre>	(trans-
key_strip()	(trans-	late.lang.code_or.code	,
late.storage.properties.DialectSkype	class	360	_or class memoa),
method), 590		length_difference()	(trans-
key_strip()	(trans-	late.lang.common.Com	,
late.storage.properties.DialectStrings	class	362	,,
method), 591		length_difference() (tra	anslate.lang.de.de class
key_strip()	(trans-	method), 366	Ü
late.storage.properties.DialectStrings class method), 591	Utf8		anslate.lang.el.el class
key_strip()	(trans-		anslate.lang.es.es class
late.storage.properties.DialectXWiki	class	method), 367	ansidic.idng.cs.cs cidss
method), 592			anslate.lang.fa.fa class
keys()(translate.misc.dictutils.cidict method), 389	method), 368	
keys() (translate.storage.oo.unormalizechar	method),	length_difference() (t	ranslate.lang.fi.fi class
khmerpunc (translate.lang.km.km attribute),	375	<pre>method), 369 length_difference() (to </pre>	anslate.lang.fr.fr class
km (class in translate.lang.km), 374		method), 370	anslate.lang.fr.fr class
kn (class in translate.lang.kn), 375			ınslate.lang.gu.gu class
ko (class in translate.lang.ko), 376		method), 371	

<pre>length_difference() (translate.lang.he.he class method), 371</pre>	s load() (translate.storage.projstore.ProjectStore method), 584		
<pre>length_difference() (translate.lang.hi.hi clas method), 372</pre>	s localName (translate.misc.ourdom.Document at- tribute), 397		
<pre>length_difference() (translate.lang.hy.hy clas method), 373</pre>	s localName (translate.misc.ourdom.Element attribute), 397		
<pre>length_difference() (translate.lang.ja.ja clas method), 374</pre>	s long() (translate.filters.checks.CCLicenseChecker method), 275		
<pre>length_difference() (translate.lang.km.km clas</pre>	method), 281		
<pre>length_difference() (translate.lang.kn.kn clas method), 375</pre>	method), 287		
length_difference() (translate.lang.ko.ko clasmethod), 376	293		
<pre>length_difference() (translate.lang.ml.ml clas method), 377</pre>	298		
length_difference() (translate.lang.mr.mr clasmethod), 378	304		
length_difference() (translate.lang.ne.ne clasmethod), 378	method), 310		
length_difference() (translate.lang.pa.pa clasmethod), 379	method), 316		
length_difference() (translate.lang.si.si clas method), 381	method), 322		
length_difference() (translate.lang.st.st class method), 381	method), 327		
<pre>length_difference() (translate.lang.sv.sv clas</pre>	method), 333		
<pre>length_difference() (translate.lang.ta.ta clas method), 383 length_difference() (translate.lang.te.te clas</pre>	method), 339		
method), 384 length_difference() (translate.lang.th.th class	346		
method), 385 length_difference() (translate.lang.ug.ug class	method), 391		
method), 386 length_difference() (translate.lang.ur.ur class	lstrip() (translate.misc.multistring.multistring		
method), 386 length_difference() (translate.lang.vi.vi class	N.4		
method), 387 length_difference() (translate.lang.zh.zh clas	main() (in module translate.tools.phppo2pypo), 714		
method), 388 LibreOfficeChecker (class in trans	main() (in module translate.tools.pypo2phppo), 731		
late.filters.checks), 308 LISAfile (class in translate.storage.lisa), 482	make_postore_adder() (in module trans- late.storage.xml_extract.extract), 711		
LISAunit (class in translate.storage.lisa), 484 488			
listseperator (translate.lang.common.Common at tribute), 362	method), 569		
listsubfiles() (translate.storage.oo.oomultifile makeheader() (translate.storage.poheader.poheader.method), 507 (translate.storage.oo.oomultifile makeheader() (translate.storage.poheader.pohea			
ljust() (translate.misc.multistring.multistring method), 391	method). 577		
load () (translate.storage.bundleprojstore.BundleProje method), 414	ctStatekeheader () (translate.storage.pypo.pofile method), 623		

makeheaderdict() (translate.storage.mo.mofile method), 489	makeindex() (translate.storage.omegat.OmegaTFile method), 499
makeheaderdict() (trans-	makeindex() (trans-
late.storage.pocommon.pofile method), 569	late.storage.omegat.OmegaTFileTab method),
makeheaderdict() (trans-	501
late.storage.poheader.poheader method),	makeindex() (translate.storage.php.LaravelPHPFile
574	method), 559
makeheaderdict() (trans-	makeindex() (translate.storage.php.phpfile method), 564
late.storage.poxliff.PoXliffFile method), 577 makeheaderdict() (translate.storage.pypo.pofile	makeindex() (translate.storage.pocommon.pofile
method), 623	method), 569
makeindex() (translate.convert.csv2po.csv2po method), 245	makeindex() (translate.storage.poxliff.PoXliffFile method), 578
makeindex() (translate.storage.base.DictStore	makeindex() (translate.storage.properties.gwtfile
method), 404	method), 596
makeindex() (translate.storage.base.TranslationStore	makeindex() (translate.storage.properties.javafile
method), 409	method), 599
makeindex() (translate.storage.catkeys.CatkeysFile	makeindex() (trans-
method), 416	late.storage.properties.javautf16file method),
makeindex() (translate.storage.csvl10n.csvfile	600
method), 421	makeindex() (translate.storage.properties.javautf8file
makeindex() (translate.storage.dtd.dtdfile method),	method), 602
428	makeindex() (translate.storage.properties.joomlafile method), 604
makeindex() (translate.storage.html.htmlfile method), 437	makeindex() (translate.storage.properties.propfile
makeindex() (translate.storage.html.POHTMLParser	method), 606
method), 434	makeindex() (translate.storage.properties.stringsfile
makeindex() (translate.storage.ical.icalfile method),	method), 614
442	makeindex() (trans-
makeindex() (translate.storage.ini.inifile method),	late.storage.properties.stringsutf8file method),
447	615
makeindex() (trans-	makeindex() (translate.storage.properties.xwikifile
late.storage.jsonl10n.ARBJsonFile method),	method), 617
453 makeindex() (trans-	makeindex() (trans-
late.storage.jsonl10n.GoI18NJsonFile	late.storage.properties.XWikiFullPage method), 593
method), 458	makeindex() (trans-
<pre>makeindex() (translate.storage.jsonl10n.I18NextFile</pre>	late.storage.properties.XWikiPageProperties method), 594
makeindex() (translate.storage.jsonl10n.JsonFile	makeindex() (translate.storage.pypo.pofile method),
method), 468	623
makeindex() (trans-	makeindex() (translate.storage.qm.qmfile method),
late.storage.jsonl10n.JsonNestedFile method),	629
469	<pre>makeindex() (translate.storage.qph.QphFile method),</pre>
makeindex() (trans-	635
late. storage. js on 110 n. Web Extension Js on File	makeindex() (translate.storage.rc.rcfile method), 641
method), 478	makeindex() (trans-
$\verb makeindex() (translate.storage.lisa.LISA file method),$	late. storage. subtitles. Adv Sub Station Alpha File
483	method), 647
<pre>makeindex() (translate.storage.mo.mofile method),</pre>	makeindex() (trans-
489	late.storage.subtitles.MicroDVDFile method),
makeindex() (trans-	649
late.storage.mozilla_lang.LangStore method), 494	makeindex() (translate.storage.subtitles.SubRipFile method), 651

makeindex() (trans-	472
late. storage. subtitles. SubStation Alpha File	makeobsolete() (trans-
method), 653	late.storage.jsonl10n.JsonUnit method),
<pre>makeindex() (translate.storage.subtitles.SubtitleFile</pre>	476
method), 655	makeobsolete() (trans-
makeindex() (translate.storage.tbx.tbxfile method), 660	late.storage.jsonl10n.WebExtensionJsonUnit method), 480
makeindex() (translate.storage.tiki.TikiStore method), 665	makeobsolete() (translate.storage.lisa.LISAunit method), 486
makeindex() (translate.storage.tmx.tmxfile method), 670	makeobsolete() (translate.storage.mo.mouning method), 492
makeindex() (trans-	makeobsolete() (trans-
late.storage.trados.TradosTxtTmFile method), 679	late.storage.mozilla_lang.LangUnit method), 497
makeindex() (translate.storage.ts2.tsfile method), 682	makeobsolete() (trans-
<pre>makeindex() (translate.storage.txt.TxtFile method), 687</pre>	late.storage.omegat.OmegaTUnit method), 504
<pre>makeindex() (translate.storage.utx.UtxFile method),</pre>	makeobsolete() (trans-
692	late.storage.php.LaravelPHPUnit method),
makeindex() (trans-	562
late.storage.wordfast.WordfastTMFile method), 699	makeobsolete() (translate.storage.php.phpunit method), 567
<pre>makeindex() (translate.storage.xliff.xlifffile method),</pre>	makeobsolete() (trans-
706	late.storage.pocommon.pounit method),
makekey() (in module translate.storage.oo), 506	572
makeobsolete() (translate.storage.base.DictUnit method), 407	makeobsolete() (trans- late.storage.poxliff.PoXliffUnit method).
makeobsolete() (trans-	582
late.storage.base.TranslationUnit method),	makeobsolete() (trans-
412	late.storage.properties.proppluralunit method),
makeobsolete() (trans-	608
late.storage.catkeys.CatkeysUnit method), 419	makeobsolete() (trans- late.storage.properties.propunit method).
<pre>makeobsolete() (translate.storage.csvl10n.csvunit</pre>	611
method), 424	makeobsolete() (trans-
makeobsolete() (translate.storage.dtd.dtdunit method), 431	late.storage.properties.xwikiunit method), 620
makeobsolete() (translate.storage.html.htmlunit method), 440	makeobsolete() (translate.storage.pypo.pounit method), 626
makeobsolete() (translate.storage.ical.icalunit method), 445	makeobsolete() (translate.storage.qm.qmunit method), 632
makeobsolete() (translate.storage.ini.iniunit method), 450	makeobsolete() (translate.storage.qph.QphUnit
makeobsolete() (trans-	makeobsolete() (translate.storage.rc.rcunii
late.storage.jsonl10n.ARBJsonUnit method),	method), 644
456	makeobsolete() (trans-
makeobsolete() (trans-	late.storage.subtitles.SubtitleUnit method),
late. storage. js on l10 n. Go I18 NJ son Unit	657
method), 461	makeobsolete() (translate.storage.tbx.tbxunii
makeobsolete() (trans-	method), 663
late.storage.jsonl10n.I18NextUnit method), 466	makeobsolete() (translate.storage.tiki.TikiUnit method), 668
<pre>makeobsolete() (trans- late.storage.jsonl10n.JsonNestedUnit method),</pre>	makeobsolete() (translate.storage.tmx.tmxunit method), 673

makeobsolete() (trans- late.storage.trados.TradosUnit method),	map() (translate.storage.placeables.strelem.StringElem method), 538
678 makeobsolete() (translate.storage.ts2.tsunit	map () (translate.storage.placeables.terminology.TerminologyPlaceable method), 540
method), 685	map() (translate.storage.placeables.xliff.Bpt method),
makeobsolete() (translate.storage.txt.TxtUnit method), 690	map() (translate.storage.placeables.xliff.Bx method),
makeobsolete() (translate.storage.utx.UtxUnit method), 696	547 map() (translate.storage.placeables.xliff.Ept method),
makeobsolete() (trans-	544
late.storage.wordfast.WordfastUnit method), 702	map() (translate.storage.placeables.xliff.Ex method), 549
makeobsolete() (translate.storage.xliff.xliffunit method), 709	map() (translate.storage.placeables.xliff.G method), 550 map() (translate.storage.placeables.xliff.It method), 552
maketrans() (translate.misc.multistring.multistring	map() (translate.storage.placeables.xliff.Ph method), 555
static method), 392 ManHelpFormatter (class in trans-	map() (translate.storage.placeables.xliff.Sub method),
late.misc.optrecurse), 393 ManPageOption (class in translate.misc.optrecurse),	554 map() (translate.storage.placeables.xliff.UnknownXML
393	method), 557
map() (translate.storage.placeables.base.Bpt method), 509	map() (translate.storage.placeables.xliff.X method), 546 markapproved() (trans-
map() (translate.storage.placeables.base.Bx method), 517	late.storage.poxliff.PoXliffUnit method), 582
map() (translate.storage.placeables.base.Ept method), 511	markapproved() (translate.storage.xliff.xliffunit method), 709
map() (translate.storage.placeables.base.Ex method), 519	markfuzzy() (translate.storage.base.DictUnit method), 407
map() (translate.storage.placeables.base.G method), 516	markfuzzy() (translate.storage.base.TranslationUnit method), 413
map() (translate.storage.placeables.base.It method), 514	markfuzzy() (translate.storage.catkeys.CatkeysUnit method), 419
map() (translate.storage.placeables.base.Ph method), 513	markfuzzy() (translate.storage.csvl10n.csvunit method), 424
map() (translate.storage.placeables.base.Sub method), 522	markfuzzy() (translate.storage.dtd.dtdunit method), 431
<pre>map() (translate.storage.placeables.base.X method),</pre>	markfuzzy() (translate.storage.html.htmlunit method), 440
map () (translate.storage.placeables.general.AltAttrPlace method), 524	ablerkfuzzy() (translate.storage.ical.icalunit method), 445
map () (translate.storage.placeables.general.XMLEntityP method), 526	lameabke uzzy () (translate.storage.ini.iniunit method), 450
map() (translate.storage.placeables.general.XMLTagPlaceables), 527	cemblekfuzzy() (trans- late.storage.jsonl10n.ARBJsonUnit method),
map () (translate.storage.placeables.interfaces.BasePlace method), 529	.
map () (translate.storage.placeables.interfaces.InvisibleP method), 531	
	lameabke uzzy () (translate.storage.jsonl10n.118NextUnit method), 466
map () (translate.storage.placeables.interfaces.Replacement method), 534	
map() (translate.storage.placeables.interfaces.SubflowPl method), 536	
тетои), 550	malatalazy () (namancaniage, jounton asonomi

method), 476	<pre>markfuzzy() (translate.storage.xliff.xliffunit method),</pre>
markfuzzy() (trans-	710
late.storage.jsonl10n.WebExtensionJsonUnit method), 481	markreviewneeded() (trans- late.storage.base.DictUnit method), 407
	markreviewneeded() (trans-
method), 486	late.storage.base.TranslationUnit method),
markfuzzy() (translate.storage.mo.mounit method),	413
492	markreviewneeded() (trans-
markfuzzy() (trans-	late.storage.catkeys.CatkeysUnit method),
late.storage.mozilla_lang.LangUnit method),	419
497	markreviewneeded() (trans-
markfuzzy() (translate.storage.omegat.OmegaTUnit	late.storage.csvl10n.csvunit method), 425
method), 504	markreviewneeded() (translate.storage.dtd.dtdunit
markfuzzy() (translate.storage.php.LaravelPHPUnit	method), 431
method), 562	markreviewneeded() (trans-
markfuzzy() (translate.storage.php.phpunit method),	late.storage.html.htmlunit method), 440
567	markreviewneeded() (trans-
markfuzzy() (translate.storage.pocommon.pounit	late.storage.ical.icalunit method), 445
method), 572	markreviewneeded() (translate.storage.ini.iniunit
markfuzzy() (translate.storage.poxliff.PoXliffUnit	method), 450
method), 582	markreviewneeded() (trans-
markfuzzy() (trans-	late.storage.jsonl10n.ARBJsonUnit method),
late.storage.properties.proppluralunit method),	456
609	markreviewneeded() (trans-
markfuzzy() (translate.storage.properties.propunit	late.storage.jsonl10n.GoI18NJsonUnit
method), 612	method), 461
	markreviewneeded() (trans-
method), 620	late.storage.jsonl10n.I18NextUnit method),
markfuzzy() (translate.storage.pypo.pounit method),	466
626	markreviewneeded() (trans-
markfuzzy() (translate.storage.qm.qmunit method),	late.storage.jsonl10n.JsonNestedUnit method),
632	473
markfuzzy() (translate.storage.qph.QphUnit	markreviewneeded() (trans-
method), 638	late.storage.jsonl10n.JsonUnit method),
markfuzzy() (translate.storage.rc.rcunit method), 644	476
	markreviewneeded() (trans-
method), 657	late.storage.jsonl10n.WebExtensionJsonUnit
markfuzzy() (translate.storage.tbx.tbxunit method),	method), 481
663	markreviewneeded() (trans-
markfuzzy() (translate.storage.tiki.TikiUnit method),	late.storage.lisa.LISAunit method), 486
668	markreviewneeded() (translate.storage.mo.mounit
markfuzzy() (translate.storage.tmx.tmxunit method),	method), 492
673	markreviewneeded() (trans-
markfuzzy() (translate.storage.trados.TradosUnit	late.storage.mozilla_lang.LangUnit method),
method), 678	497
markfuzzy() (translate.storage.ts2.tsunit method),	markreviewneeded() (trans-
685	late.storage.omegat.OmegaTUnit method),
<pre>markfuzzy() (translate.storage.txt.TxtUnit method),</pre>	504
690	markreviewneeded() (trans-
markfuzzy() (translate.storage.utx.UtxUnit method),	late.storage.php.LaravelPHPUnit method),
696	562
markfuzzy() (trans-	markreviewneeded() (trans-
late.storage.wordfast.WordfastUnit method),	late.storage.php.phpunit method), 567
702	markreviewneeded() (trans-

late.storage.pocommon.pounit method),	match_source() (in module trans-
573	late.tools.pretranslate), 729
markreviewneeded() (trans-	match_template_id() (in module trans-
late.storage.poxliff.PoXliffUnit method),	late.tools.pretranslate), 729
582 markreviewneeded() (trans-	match_template_location() (in module trans-
markreviewneeded() (trans- late.storage.properties.proppluralunit method),	late.tools.pretranslate), 729 matcher (class in translate.search.match), 401
609	matchers (translate.storage.placeables.terminology.TerminologyPlaceables.terminologyPlac
markreviewneeded() (trans-	attribute), 540
late.storage.properties.propunit method),	
612	402
markreviewneeded() (trans-	matches()(translate.search.match.terminologymatcher
late.storage.properties.xwikiunit method),	method), 402
620	memory () (in module translate.tools.pretranslate), 729
markreviewneeded() (trans-	merge() (translate.storage.base.DictUnit method), 407
late.storage.pypo.pounit method), 626	merge() (translate.storage.base.TranslationUnit
markreviewneeded() (translate.storage.qm.qmunit	method), 413
method), 632	merge() (translate.storage.catkeys.CatkeysUnit
markreviewneeded() (trans-	method), 419
late.storage.qph.QphUnit method), 638 markreviewneeded() (translate.storage.rc.rcunit	merge() (translate.storage.csvl10n.csvunit method), 425
method), 644	merge() (translate.storage.dtd.dtdunit method), 431
markreviewneeded() (trans-	
late.storage.subtitles.SubtitleUnit method),	3 11 (
658	merge() (translate.storage.ini.iniunit method), 450
markreviewneeded() (translate.storage.tbx.tbxunit	
method), 663	method), 456
markreviewneeded() (trans- late.storage.tiki.TikiUnit method), 668	merge() (translate.storage.jsonl10n.GoI18NJsonUnit method), 461
markreviewneeded() (trans-	merge() (translate.storage.jsonl10n.I18NextUnit
late.storage.tmx.tmxunit method), 673	method), 466
markreviewneeded() (trans-	merge() (translate.storage.jsonl10n.JsonNestedUnit
late.storage.trados.TradosUnit method),	method), 473
678	merge() (translate.storage.jsonl10n.JsonUnit method),
<pre>markreviewneeded() (translate.storage.ts2.tsunit</pre>	476
method), 685	merge() (translate.storage.jsonl10n.WebExtensionJsonUnit
markreviewneeded() (translate.storage.txt.TxtUnit	method), 481
method), 690	merge() (translate.storage.lisa.LISAunit method), 486
markreviewneeded() (trans- late.storage.utx.UtxUnit method), 696	merge() (translate.storage.mo.mounit method), 492 merge() (translate.storage.mozilla_lang.LangUnit
markreviewneeded() (trans-	method), 497
late.storage.wordfast.WordfastUnit method),	merge() (translate.storage.omegat.OmegaTUnit
702	method), 504
markreviewneeded() (trans-	merge() (translate.storage.php.LaravelPHPUnit
late.storage.xliff.xliffunit method), 710	method), 562
MaskingPlaceable (class in trans-	merge() (translate.storage.php.phpunit method), 567
late.storage.placeables.interfaces), 531	merge() (translate.storage.pocommon.pounit method),
<pre>match_entities() (trans-</pre>	573
late.convert.accesskey.UnitMixer method),	merge() (translate.storage.poxliff.PoXliffUnit method),
236	582
match_fuzzy() (in module trans- late.tools.pretranslate), 729	merge() (translate.storage.properties.proppluralunit method), 609
match_header() (translate.storage.csvl10n.csvunit	·
method), 425	method), 612

(translate.storage.properties.xwikiunit attribute), 501 merge() method), 620 merge on (translate.storage.php.LaravelPHPFile atmerge () (translate.storage.pypo.pounit method), 627 *tribute*), 559 merge() (translate.storage.qm.qmunit method), 633 merge_on (translate.storage.php.phpfile attribute), 564 merge() (translate.storage.qph.QphUnit method), 638 merge on (translate.storage.pocommon.pofile merge() (translate.storage.rc.rcunit method), 644 tribute), 569 (translate.storage.subtitles.SubtitleUnit (translate.storage.poxliff.PoXliffFile merge() merge_on at*method*), 658 *tribute*), 578 merge() (translate.storage.tbx.tbxunit method), 663 merge_on (translate.storage.properties.gwtfile atmerge() (translate.storage.tiki.TikiUnit method), 668 *tribute*), 596 merge() (translate.storage.tmx.tmxunit method), 674 merge_on (translate.storage.properties.javafile atmerge() (translate.storage.trados.TradosUnit method), tribute), 599 merge_on (translate.storage.properties.javautf16file atmerge() (translate.storage.ts2.tsunit method), 685 tribute), 600 merge() (translate.storage.txt.TxtUnit method), 690 merge_on (translate.storage.properties.javautf8file atmerge () (translate.storage.utx.UtxUnit method), 696 tribute), 602 (translate.storage.wordfast.WordfastUnit merge_on (translate.storage.properties.joomlafile atmerge() method), 703 tribute), 604 merge() (translate.storage.xliff.xliffunit method), 710 (translate.storage.properties.propfile merge_on merge on (translate.storage.base.DictStore attribute), *tribute*), 606 404 merge_on (translate.storage.properties.stringsfile atmerge_on (translate.storage.base.TranslationStore attribute), 614 tribute), 409 merge_on (translate.storage.properties.stringsutf8file merge_on (translate.storage.catkeys.CatkeysFile attribute), 615 merge_on (translate.storage.properties.xwikifile atattribute), 416 merge_on (translate.storage.csvl10n.csvfile attribute), tribute), 617 merge_on (translate.storage.properties.XWikiFullPage merge_on (translate.storage.dtd.dtdfile attribute), 428 attribute), 593 merge_on (translate.storage.html.htmlfile attribute), merge_on (translate.storage.properties.XWikiPageProperties 437 attribute), 594 merge_on (translate.storage.html.POHTMLParser atmerge_on (translate.storage.pypo.pofile attribute), 623 tribute), 434 merge_on (translate.storage.qm.qmfile attribute), 629 merge_on (translate.storage.ical.icalfile attribute), 442 merge_on (translate.storage.qph.QphFile attribute), merge_on (translate.storage.ini.inifile attribute), 447 635 merge on (translate.storage.jsonl10n.ARBJsonFile atmerge on (translate.storage.rc.rcfile attribute), 641 tribute), 453 merge on (translate.storage.subtitles.AdvSubStationAlphaFile merge on (translate.storage.jsonl10n.GoI18NJsonFile attribute), 647 attribute), 458 (translate. storage. subtitles. Micro DVDFilemerge_on merge_on (translate.storage.jsonl10n.118NextFile atattribute), 649 tribute), 463 merge_on (translate.storage.subtitles.SubRipFile atmerge_on (translate.storage.jsonl10n.JsonFile tribute), 651 tribute), 468 merge_on (translate.storage.subtitles.SubStationAlphaFile merge_on (translate.storage.jsonl10n.JsonNestedFile attribute), 653 attribute), 469 merge_on (translate.storage.subtitles.SubtitleFile atmerge_on(translate.storage.jsonl10n.WebExtensionJsonFile *tribute*), 655 attribute), 478 merge_on (translate.storage.tbx.tbxfile attribute), 660 (translate.storage.lisa.LISAfile attribute), merge_on (translate.storage.tiki.TikiStore attribute), merge_on 665 483 merge_on (translate.storage.mo.mofile attribute), 489 merge_on (translate.storage.tmx.tmxfile attribute), 670 merge_on (translate.storage.mozilla_lang.LangStore (translate.storage.trados.TradosTxtTmFilemerge_on attribute), 494 attribute), 679 merge_on (translate.storage.omegat.OmegaTFile atmerge_on (translate.storage.ts2.tsfile attribute), 682 *tribute*), 499 merge on (translate.storage.txt.TxtFile attribute), 687 merge on (translate.storage.utx.UtxFile attribute), 693 (translate.storage.omegat.OmegaTFileTab

<pre>merge_on (translate.storage.wordfast.WordfastTMFile</pre>	648
attribute), 699	Mimetypes (translate.storage.base.TranslationStore at-
merge_on (translate.storage.xliff.xlifffile attribute), 706	tribute), 408
<pre>merge_store() (in module translate.convert.sub2po),</pre>	MinimalChecker (class in translate.filters.checks),
269	314
merge_store() (translate.convert.json2po.json2po method), 248	miscpunc (translate.lang.common.Common attribute), 363
merge_store() (translate.convert.rc2po.rc2po method), 269	<pre>mix_units() (translate.convert.accesskey.UnitMixer</pre>
merge_store() (translate.convert.resx2po.resx2po method), 269	mkdir() (translate.convert.convert.ArchiveConvertOptionParser method), 239
merge_stores() (translate.convert.ical2po.ical2po method), 246	mkdir() (translate.convert.convert.ConvertOptionParser method), 242
merge_stores() (translate.convert.ini2po.ini2po method), 247	mkdir() (translate.convert.po2moz.MozConvertOptionParser method), 254
merge_stores() (trans-	mkdir() (translate.convert.po2tmx.TmxOptionParser
late.convert.mozlang2po.lang2po method),	method), 260
249	mkdir() (translate.convert.po2wordfast.WfOptionParser
<pre>merge_stores() (translate.convert.php2po.php2po</pre>	method), 264
method), 250	mkdir() (translate.filters.pofilter.FilterOptionParser
<pre>merge_stores() (translate.convert.po2ical.po2ical</pre>	method), 354
method), 251	mkdir()(translate.misc.optrecurse.RecursiveOptionParser
<pre>merge_stores() (translate.convert.po2ini.po2ini</pre>	method), 395
method), 252	mkdir()(translate.tools.poconflicts.ConflictOptionParser
<pre>merge_stores() (translate.convert.po2txt.po2txt</pre>	method), 717
method), 262	mkdir() (translate.tools.pogrep.GrepOptionParser
merge_stores() (trans-	method), 720
late.convert.po2yaml.po2yaml method),	mkdir()(translate.tools.porestructure.SplitOptionParser
267	method), 724
<pre>merge_stores() (translate.convert.txt2po.txt2po</pre>	mkdir()(translate.tools.poterminology.TerminologyOptionParser
method), 271	method), 728
merge_stores() (trans-	ml (class in translate.lang.ml), 377
late.convert.yaml2po.yaml2po method),	mofile (class in translate.storage.mo), 487
272	mounit (class in translate.storage.mo), 490
mergeheaders() (translate.storage.mo.mofile	mounpack () (in module translate.storage.mo), 493
method), 489	MozConvertOptionParser (class in trans-
mergeheaders() (translate.storage.pocommon.pofile	late.convert.po2moz), 253
method), 569	mozilla_pluralequation (trans-
mergeheaders() (trans-	late.lang.common.Common attribute), 363
late.storage.poheader.poheader method), 574	MozillaChecker (class in translate.filters.checks), 319
mergeheaders () (translate.storage.poxliff.PoXliffFile method), 578	mozillaescapemarginspaces() (in module translate.misc.quote), 399
mergeheaders() (translate.storage.pypo.pofile	mr (class in translate.lang.mr), 377
method), 623	msgidcomment (translate.storage.pypo.pounit at-
<pre>mergestore() (translate.convert.po2html.po2html</pre>	tribute), 627
method), 251	multifilter() (in module translate.filters.helpers),
<pre>mergestore() (translate.convert.prop2po.prop2po</pre>	352
method), 268	multifiltertestmethod() (in module trans-
<pre>mergestores() (in module translate.tools.pomerge),</pre>	late.filters.helpers), 352
722	multistring (class in translate.misc.multistring), 389
MessageProgressBar (class in trans-	multistring_to_rich() (trans-
late.misc.progressbar), 398	late.storage.base.DictUnit method), 407
MicroDVDFile (class in translate.storage.subtitles),	multistring to rich() (trans-

late.storage.base.TranslationUnit	method),	multistring_to_rich()	(trans-
	(4	late.storage.properties.proppluralunit	тетоа),
multistring_to_rich()	(trans-	609	(1
late.storage.catkeys.CatkeysUnit	method),	multistring_to_rich()	(trans-
419	(4	late.storage.properties.propunit	method),
multistring_to_rich()	(trans-	612	(4
late.storage.csvl10n.csvunit method),		multistring_to_rich()	(trans-
<pre>multistring_to_rich() late.storage.dtd.dtdunit method), 431</pre>	(trans-	late.storage.properties.xwikiunit 620	method),
<pre>multistring_to_rich()</pre>	(trans-	<pre>multistring_to_rich()</pre>	(trans-
late.storage.html.htmlunit method), 4	40	late.storage.pypo.pounit method), 627	7
<pre>multistring_to_rich()</pre>	(trans-	<pre>multistring_to_rich()</pre>	(trans-
late.storage.ical.icalunit method), 44	5	late.storage.qm.qmunit method), 633	
<pre>multistring_to_rich()</pre>	(trans-	<pre>multistring_to_rich()</pre>	(trans-
late.storage.ini.iniunit method), 450		late.storage.qph.QphUnit method), 63	38
<pre>multistring_to_rich()</pre>	(trans-	<pre>multistring_to_rich()</pre>	(trans-
late.storage.jsonl 10 n. ARB Json Unit	method),	late.storage.rc.rcunit method), 644	
456		<pre>multistring_to_rich()</pre>	(trans-
<pre>multistring_to_rich() late.storage.jsonl10n.GoI18NJsonUn</pre>	(trans-	late.storage.subtitles.SubtitleUnit 658	method),
method), 461		<pre>multistring_to_rich()</pre>	(trans-
multistring_to_rich()	(trans-	late.storage.tbx.tbxunit method), 663	(ii cii is
late.storage.jsonl10n.I18NextUnit	method),	multistring_to_rich()	(trans-
466	,	late.storage.tiki.TikiUnit method), 668	3
<pre>multistring_to_rich()</pre>	(trans-	<pre>multistring_to_rich()</pre>	(trans-
late.storage.jsonl10n.JsonNestedUni	t method),	late.storage.tmx.tmxunit method), 674	
473		<pre>multistring_to_rich()</pre>	(trans-
<pre>multistring_to_rich()</pre>	(trans-	late. storage. trados. Trados Unit	method),
late.storage.jsonl 10 n. Json Unit	method),	678	
476		<pre>multistring_to_rich()</pre>	(trans-
<pre>multistring_to_rich()</pre>	(trans-	late.storage.ts2.tsunit method), 685	
late.storage.jsonl10n.WebExtensionJ. method), 481	sonUnit	<pre>multistring_to_rich() late.storage.txt.TxtUnit method), 690</pre>	(trans-
multistring_to_rich()	(trans-	multistring_to_rich()	(trans-
late.storage.lisa.LISAunit method), 4	*	late.storage.utx.UtxUnit method), 696	•
multistring_to_rich()		<pre>multistring_to_rich()</pre>	(trans-
late.storage.mo.mounit method), 492		late.storage.wordfast.WordfastUnit	method),
<pre>multistring_to_rich()</pre>	(trans-	703	, ,
late.storage.mozilla_lang.LangUnit	method),	<pre>multistring_to_rich()</pre>	(trans-
497	,,,,	late.storage.xliff.xliffunit class	method),
<pre>multistring_to_rich()</pre>	(trans-	710	,
late.storage.omegat.OmegaTUnit	method),	musttranslatewords()	(trans-
504	memoa),	late.filters.checks.CCLicenseChecker	
multistring_to_rich()	(trans-	275	memou),
late.storage.php.LaravelPHPUnit	method),	musttranslatewords()	(trans-
562	memoa),	late.filters.checks.DrupalChecker	method),
multistring_to_rich()	(trans-	281	memoa),
late.storage.php.phpunit method), 56	`	musttranslatewords()	(trans-
multistring_to_rich()	(trans-	late.filters.checks.GnomeChecker	method),
late.storage.pocommon.pounit	method),	287	тетои),
573	memou),		(trans
	(trans	musttranslatewords() late.filters.checks.IOSChecker method	(trans-
multistring_to_rich()	(trans-		
late.storage.poxliff.PoXliffUnit 582	method),	<pre>musttranslatewords() late.filters.checks.KdeChecker</pre>	(trans- method),

299	<pre>namespaced() (translate.storage.ts2.tsunit method),</pre>
musttranslatewords() (trans-	685
late.filters.checks.L20nChecker method), 304	namespaced() (translate.storage.xliff.xlifffile method), 706
musttranslatewords() (trans-	namespaced() (translate.storage.xliff.xliffunit
late.filters.checks.LibreOfficeChecker method), 310	<pre>method), 710 native_distance() (in module trans-</pre>
musttranslatewords() (trans-	late.search.lshtein), 401
late.filters.checks.MinimalChecker method),	ne (class in translate.lang.ne), 378
316	newlines () (translate.filters.checks.CCLicenseChecker
musttranslatewords() (trans-	method), 275
late.filters.checks.MozillaChecker method), 322	newlines() (translate.filters.checks.DrupalChecker method), 281
musttranslatewords() (trans-	newlines() (translate.filters.checks.GnomeChecker
late.filters.checks.OpenOfficeChecker method),	method), 287
327	newlines() (translate.filters.checks.IOSChecker
musttranslatewords() (trans-	method), 293
late.filters.checks.ReducedChecker method),	newlines() (translate.filters.checks.KdeChecker
333	method), 299
musttranslatewords() (trans-	newlines() (translate.filters.checks.L20nChecker
late.filters.checks.StandardChecker method),	method), 305
339	newlines () (translate.filters.checks.LibreOfficeChecker
musttranslatewords() (trans-	method), 310
late.filters.checks.TermChecker method),	newlines() (translate.filters.checks.MinimalChecker
346	method), 316
N	newlines() (translate.filters.checks.MozillaChecker method), 322
Name (translate.storage.base.TranslationStore attribute), 408	newlines() (translate.filters.checks.OpenOfficeChecker method), 327
namespaced() (in module trans- late.misc.xml_helpers), 400	newlines() (translate.filters.checks.ReducedChecker method), 333
namespaced() (translate.storage.lisa.LISAfile method), 483	newlines () (translate.filters.checks.StandardChecker method), 339
namespaced() (translate.storage.lisa.LISAunit method), 486	newlines () (translate.filters.checks.TermChecker method), 346
namespaced() (translate.storage.poxliff.PoXliffFile	NoInitialStateError,704
method), 578 namespaced() (translate.storage.poxliff.PoXliffUnit	NoProgressBar (class in translate.misc.progressbar), 398
method), 582	normalize() (in module translate.lang.data), 365
namespaced() (translate.storage.qph.QphFile	normalize_space() (in module trans-
method), 635	late.misc.xml_helpers), 400
namespaced() (translate.storage.qph.QphUnit method), 638	normalize_xml_space() (in module trans- late.misc.xml_helpers), 400
namespaced() (translate.storage.tbx.tbxfile method),	normalized_unicode() (in module trans-
660	late.lang.data), 365
namespaced() (translate.storage.tbx.tbxunit method), 663	normalizefilename() (in module trans- late.storage.oo), 506
namespaced() (translate.storage.tmx.tmxfile method),	notranslatewords() (trans-
namespaced() (translate.storage.tmx.tmxunit	late.filters.checks.CCLicenseChecker method), 275
method), 674	notranslatewords() (trans-
namespaced() (translate.storage.ts2.tsfile method), 682	late.filters.checks.DrupalChecker method), 281
	notranslatewords() (trans-

late.filters.checks.GnomeChecker m 287	nethod),	method), 328 numbers() (translat	e.filters.checks.ReducedC	hockor
notranslatewords()	(trans-	method), 333		
<pre>late.filters.checks.IOSChecker method), notranslatewords()</pre>	(trans-	numbers() (translate method), 339	e.filters.checks.StandardC	пескег
	ethod),		slate.filters.checks.TermC	hecker
299	,,	method), 346	J	
notranslatewords()	(trans-	<pre>numbertranslate()</pre>	(translate.lang.af.af	class
· ·	iethod),	method), 357		
305		numbertranslate()	(translate.lang.am.am	class
notranslatewords()	(trans-	method), 358	(4	.1
late.filters.checks.LibreOfficeChecker m 310	ieinoa),	numbertranslate() <i>method</i>), 359	(translate.lang.ar.ar	class
notranslatewords()	(trans-	numbertranslate()	(translate.lang.bn.bn	class
	nethod),	method), 359	(iranstate.tang.on.on	crass
316	,,	numbertranslate()	((trans-
notranslatewords()	(trans-	late.lang.code_	or.code_or class me	ethod),
· ·	iethod),	360		
322		<pre>numbertranslate()</pre>		(trans-
notranslatewords()	(trans-	late.lang.comm	on.Common class me	ethod),
late.filters.checks.OpenOfficeChecker m 328	ietnoa),	363 numbertranslate()	(translate.lang.de.de	class
notranslatewords()	(trans-	method), 366	(transtate.tang.ae.ae	ciuss
	nethod),	numbertranslate()	(translate.lang.el.el	class
333	,,	method), 366	(
notranslatewords()	(trans-	numbertranslate()	(translate.lang.es.es	class
late.filters.checks.StandardChecker m	iethod),	method), 367		
339		<pre>numbertranslate()</pre>	(translate.lang.fa.fa	class
notranslatewords()	(trans-	<i>method</i>), 368		7
late.filters.checks.TermChecker m 346	iethod),	numbertranslate()	(translate.lang.fi.fi	class
nplurals (translate.lang.common.Common att	ribute)	<pre>method), 369 numbertranslate()</pre>	(translate.lang.fr.fr	class
363	ribuie),	method), 370	(transtate.tang.jr.jr	ciuss
nplurals()(translate.filters.checks.StandardU	nitCheck	//	(translate.lang.gu.gu	class
method), 343		method), 371		
numbers() (translate.filters.checks.CCLicense(Checker	<pre>numbertranslate()</pre>	(translate.lang.he.he	class
method), 276		method), 371		_
numbers() (translate.filters.checks.Drupal	Checker	numbertranslate()	(translate.lang.hi.hi	class
method), 282 numbers() (translate.filters.checks.Gnome(Chacker	<pre>method), 372 numbertranslate()</pre>	(translate.lang.hy.hy	class
method), 287	лескет	method), 373	(transtate.tang.ny.ny	ciuss
numbers () (translate.filters.checks.IOSO	Checker	numbertranslate()	(translate.lang.ja.ja	class
method), 293		method), 374	(* * * * * * * * * * * * * * * * * * *	
numbers() (translate.filters.checks.Kde	Checker	<pre>numbertranslate()</pre>	(translate.lang.km.km	class
method), 299		method), 375		
numbers() (translate.filters.checks.L20n0	Checker	numbertranslate()	(translate.lang.kn.kn	class
method), 305	~1 1	method), 375	(, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,	1
numbers () (translate.filters.checks.LibreOffice	лескег	numbertranslate() <i>method</i>), 376	(translate.lang.ko.ko	class
numbers() (translate.filters.checks.Minimal(Checker	numbertranslate()	(translate.lang.ml.ml	class
method), 316	ciwi	<i>method</i>), 377	(in wiscone it with the internal in the inter	CIGID
numbers () (translate.filters.checks.Mozilla	Checker	numbertranslate()	(translate.lang.mr.mr	class
method), 322		method), 378		
numbers() (translate.filters.checks.OpenOfficeC	Checker	<pre>numbertranslate()</pre>	(translate.lang.ne.ne	class

method), 378		numstart () (translate.lang.ta.ta class method), 383
		numstart () (translate.lang.te.te class method), 384
method), 379		numstart() (translate.lang.th.th class method), 385
		numstart() (translate.lang.ug.ug class method), 386
method), 381		numstart() (translate.lang.ur.ur class method), 386
		numstart() (translate.lang.vi.vi class method), 387
method), 382		numstart() (translate.lang.zh.zh class method), 388
	lass	
method), 382		O
	lass	OMEGAT_FIELDNAMES (in module trans-
method), 383		late.storage.omegat), 498
numbertranslate() (translate.lang.te.te cl	lass	OmegaTDialect (class in translate.storage.omegat),
method), 384		498
numbertranslate() (translate.lang.th.th cl	lass	OmegaTFile (class in translate.storage.omegat), 498
method), 385		OmegaTFileTab (class in translate.storage.omegat),
numbertranslate() (translate.lang.ug.ug classification)	lass	500
method), 386		OmegaTUnit (class in translate.storage.omegat), 502
numbertranslate() (translate.lang.ur.ur classical	7	oofile (class in translate.storage.oo), 506
method), 386		ooline (class in translate.storage.oo), 506
numbertranslate() (translate.lang.vi.vi classical	7	oomultifile (class in translate.storage.oo), 506
method), 387		oounit (class in translate.storage.oo), 507
numbertranslate() (translate.lang.zh.zh cla	7	openarchive() (trans-
method), 388		late.convert.convert.ArchiveConvertOptionParse
numbertuple (translate.lang.common.Comm	non	method), 239
attribute), 363		openarchive() (trans-
numstart () (translate.lang.af.af class method), 357		late.convert.po2tmx.TmxOptionParser
numstart() (translate.lang.am.am class method), 3	358	method), 260
numstart () (translate.lang.ar.ar class method), 359	9	openarchive() (trans-
numstart() (translate.lang.bn.bn class method), 35	59	late.convert.po2wordfast.WfOptionParser
<pre>numstart() (translate.lang.code_or.code_or class)</pre>	lass	method), 264
method), 360		openinputfile() (trans-
numstart() (translate.lang.common.Common cla	lass	late.convert.convert.ArchiveConvertOptionParse
method), 363		method), 239
numstart() (translate.lang.de.de class method), 36	66	openinputfile() (trans-
numstart () (translate.lang.el.el class method), 366	5	late.convert.convert.ConvertOptionParser
numstart () (translate.lang.es.es class method), 367	7	method), 242
numstart () (translate.lang.fa.fa class method), 368	8	openinputfile() (trans-
numstart() (translate.lang.fi.fi class method), 369		late.convert.po2moz.MozConvertOptionParser
$\verb numstart() \textit{(translate.lang.fr.fr class method)}, 370$)	method), 254
numstart() (translate.lang.gu.gu class method), 37	71	openinputfile() (trans-
numstart () (translate.lang.he.he class method), 37	71	late.convert.po2tmx.TmxOptionParser
numstart () (translate.lang.hi.hi class method), 372	2	method), 260
numstart() (translate.lang.hy.hy class method), 37	73	openinputfile() (trans-
numstart () (translate.lang.ja.ja class method), 374	4	late.convert.po2wordfast.WfOptionParser
numstart() (translate.lang.km.km class method), 3	375	method), 264
numstart () (translate.lang.kn.kn class method), 37	75	openinputfile() (trans-
numstart () (translate.lang.ko.ko class method), 37	76	late.filters.pofilter.FilterOptionParser method),
numstart() (translate.lang.ml.ml class method), 37		354
numstart () (translate.lang.mr.mr class method), 37	78	openinputfile() (trans-
numstart () (translate.lang.ne.ne class method), 37	78	late.misc.optrecurse.RecursiveOptionParser
numstart () (translate.lang.pa.pa class method), 37	79	method), 395
$\verb numstart() (\textit{translate.lang.si.si class method}), 381$	L	openinputfile() (translate.storage.oo.oomultifile
numstart () (translate.lang.st.st class method), 382	2	method), 507
numstart () (translate.lang.sv.sv class method), 382	2	**

openinputfile()	(trans-	method), 242	
late.tools.poconflicts.ConflictOptionP	Parser	opentemplatefile()	(trans-
method), 717		late.convert.po2moz.MozConvertOpti	onParser
openinputfile()	(trans-	method), 254	
late. to ols. pogrep. Grep Option Parser	method),	<pre>opentemplatefile()</pre>	(trans-
720		late.convert.po2tmx.TmxOptionParset	r
openinputfile()	(trans-	method), 260	
late.tools.porestructure.SplitOptionPa	ırser	<pre>opentemplatefile()</pre>	(trans-
method), 724	(4	late.convert.po2wordfast.WfOptionPa	rser
openinputfile()	(trans-	method), 265	(4
late.tools.poterminology.Terminology	OptionParse		(trans-
method), 728	,	late.filters.pofilter.FilterOptionParser	metnoa),
OpenOfficeChecker (class in	trans-	354	(4
late.filters.checks), 325	(4	opentemplatefile()	(trans-
openoutputfile()	(trans-	late.misc.optrecurse.RecursiveOption	Parser
late.convert.convert.ArchiveConvertO	ptionParsei		(4
method), 239	(4	opentemplatefile()	(trans-
openoutputfile()	(trans-	late.tools.poconflicts.ConflictOptionP	arser
late.convert.convert.ConvertOptionPa	irser	method), 717	(4
method), 242	(4	opentemplatefile()	(trans-
openoutputfile()	(trans-	late.tools.pogrep.GrepOptionParser	тетоа),
late.convert.po2moz.MozConvertOpti	onParser	721	(4
method), 254	(4	opentemplatefile()	(trans-
openoutputfile()	(trans-	late.tools.porestructure.SplitOptionPo	irser
late.convert.po2tmx.TmxOptionParse.	r	method), 724	(4
method), 260	(tuana	opentemplatefile()	(trans-
openoutputfile()	(trans-	late.tools.poterminology.Terminology	OpiionParser
late.convert.po2wordfast.WfOptionPa	irser	method), 728	(tracers
method), 264	(tuana	opentempoutputfile()	(trans-
openoutputfile()	(trans-	late.convert.convert.ArchiveConvertO	puonParser
late.filters.pofilter.FilterOptionParser 354	meinoa),	method), 239	(tracers
	(tuana	opentempoutputfile()	(trans-
openoutputfile()	(trans-	late.convert.convert.ConvertOptionPa	irser
late.misc.optrecurse.RecursiveOption method), 395	raisei	method), 242	(tuana
openoutputfile() (translate.storage.oo.d	oomultifila	<pre>opentempoutputfile() late.convert.po2moz.MozConvertOpti</pre>	(trans-
method), 507	отинуне	method), 254	oni arser
	(trans		(trans-
<pre>openoutputfile() late.tools.poconflicts.ConflictOptionP</pre>		<pre>opentempoutputfile() late.convert.po2tmx.TmxOptionParse.</pre>	,
method), 717	urser	method), 260	•
openoutputfile()	(trans-	opentempoutputfile()	(trans-
late.tools.pogrep.GrepOptionParser	method),	late.convert.po2wordfast.WfOptionPa	`
720	memou),	method), 265	1361
openoutputfile()	(trans-	opentempoutputfile()	(trans-
late.tools.porestructure.SplitOptionPomethod), 724	ırser	late.filters.pofilter.FilterOptionParser 354	method),
openoutputfile()	(trans-	opentempoutputfile()	(trans-
late.tools.poterminology.Terminology	`	er late.misc.optrecurse.RecursiveOption	`
method), 728		method), 395	
<pre>opentemplatefile()</pre>		opentempoutputfile()	(trans-
late.convert.convert.ArchiveConvertO method), 239)ptionParser	late.tools.poconflicts.ConflictOptionP method), 717	arser
opentemplatefile()	(trans-	opentempoutputfile()	(trans-
late.convert.convert.ConvertOptionPa	ırser	late. tools. pogrep. Grep Option Parser	method),

721	parse() (translate.storage.html.htmlfile method), 437
opentempoutputfile() (trans-	parse() (translate.storage.html.POHTMLParser
late. tools. por estructure. Split Option Parser	method), 434
method), 724	parse() (translate.storage.ical.icalfile method), 442
	parse() (translate.storage.ini.inifile method), 447
late.tools.poterminology.TerminologyOptionPars method), 728	separse() (translate.storage.jsonl10n.ARBJsonFile method), 453
options() (translate.filters.checks.CCLicenseChecker method), 276	parse() (translate.storage.jsonl10n.GoI18NJsonFile method), 458
options() (translate.filters.checks.DrupalChecker method), 282	
options() (translate.filters.checks.GnomeChecker method), 288	parse() (translate.storage.jsonl10n.JsonFile method), 468
options() (translate.filters.checks.IOSChecker method), 293	parse() (translate.storage.jsonl10n.JsonNestedFile method), 470
	parse() (translate.storage.jsonl10n.WebExtensionJsonFile method), 478
	parse() (translate.storage.lisa.LISAfile method), 483
method), 305	parse() (translate.storage.mo.mofile method), 489
options() (translate.filters.checks.LibreOfficeChecker method), 311	parse() (translate.storage.mozilla_lang.LangStore method), 494
options() (translate.filters.checks.MinimalChecker method), 316	parse() (translate.storage.omegat.OmegaTFile method), 499
$\verb"options"() \qquad \textit{(translate.filters.checks.MozillaChecker}$	
method), 322	method), 501
options () (translate.filters.checks.OpenOfficeChecker method), 328	parse() (translate.storage.php.LaravelPHPFile
options() (translate.filters.checks.ReducedChecker	method), 559
method), 333	parse() (translate.storage.php.phpfile method), 564
method), 339	parse() (translate.storage.placeables.general.AltAttrPlaceable class method), 524
method), 346	parse() (translate.storage.placeables.general.XMLEntityPlaceable class method), 526
	parse() (translate.storage.placeables.general.XMLTagPlaceable
late.tools.poconflicts.ConflictOptionParser method), 717	class method), 527 parse() (translate.storage.placeables.strelem.StringElem
outputterminology() (trans-	class method), 538
	separse() (translate.storage.placeables.terminology.TerminologyPlaceable class method), 540
P	parse() (translate.storage.placeables.xliff.UnknownXML class method), 557
pa (class in translate.lang.pa), 379	parse() (translate.storage.pocommon.pofile method),
parse() (in module translate.misc.ourdom), 398	569
parse() (in module trans- late.storage.placeables.parse), 536	parse() (translate.storage.poxliff.PoXliffFile method), 578
${\tt parse()} \textit{(translate.storage.base.DictStore} \textit{method)}, \\ 404$	parse() (translate.storage.properties.gwtfile method), 596
parse() (translate.storage.base.TranslationStore method), 409	parse() (translate.storage.properties.javafile method), 599
parse() (translate.storage.catkeys.CatkeysFile method), 416	parse() (translate.storage.properties.javautf16file method), 600
parse() (translate.storage.csvl10n.csvfile method), 422	parse() (translate.storage.properties.javautf8file
parse() (translate.storage.dtd.dtdfile method), 428	method), 602
parse() (translate.storage.dtd.dtdunit method), 431	parse() (translate.storage.properties.joomlafile

method), 604	late.filters.pofilter.FilterOptionParser method),
<pre>parse() (translate.storage.properties.propfile method),</pre>	354
606	parse_args() (trans-
parse() (translate.storage.properties.stringsfile	late.misc.optrecurse.RecursiveOptionParser
method), 614	method), 395
parse() (translate.storage.properties.stringsutf8file	parse_args() (trans-
method), 615	late.tools.poconflicts.ConflictOptionParser
parse() (translate.storage.properties.xwikifile	method), 717
method), 617	parse_args() (trans-
parse() (translate.storage.properties.XWikiFullPage	late.tools.pogrep.GrepOptionParser method),
method), 593	721
parse() (translate.storage.properties.XWikiPageProperti	
method), 594	late.tools.porestructure.SplitOptionParser
parse() (translate.storage.pypo.pofile method), 623	method), 724
parse () (translate.storage.qm.qmfile method), 630	parse_args() (trans-
parse() (translate.storage.qph.QphFile method), 635	late.tools.poterminology.TerminologyOptionParser
parse() (translate.storage.rc.rcfile method), 641	method), 728
parse() (translate.storage.subtitles.AdvSubStationAlpha.	
method), 648	late.storage.benchmark.TranslateBenchmarker
parse() (translate.storage.subtitles.MicroDVDFile	method), 414
method), 649	parse_noinput() (trans-
parse() (translate.storage.subtitles.SubRipFile	$late. filters. pofilter. Filter Option Parser\ method),$
method), 651	354
$\verb"parse()" (translate.storage.subtitles.SubStationAlphaFile")$	parse_placeables() (trans-
method), 653	late.storage.benchmark.TranslateBenchmarker
parse() (translate.storage.subtitles.SubtitleFile	method), 414
method), 655	<pre>parse_tag() (in module trans-</pre>
parse() (translate.storage.tbx.tbxfile method), 660	late.storage.xml_extract.misc), 712
parse() (translate.storage.tiki.TikiStore method), 665	ParseError, 408
parse() (translate.storage.tmx.tmxfile method), 671	<pre>parseFile() (translate.misc.ourdom.ExpatBuilderNS</pre>
<pre>parse() (translate.storage.trados.TradosTxtTmFile</pre>	method), 397
method), 680	parsefile() (translate.storage.base.DictStore class
parse() (translate.storage.ts2.tsfile method), 682	method), 404
parse() (translate.storage.txt.TxtFile method), 687	parsefile() (translate.storage.base.TranslationStore
parse() (translate.storage.utx.UtxFile method), 693	class method), 409
parse() (translate.storage.wordfast.WordfastTMFile	parsefile() (translate.storage.catkeys.CatkeysFile
method), 699	class method), 416
	parsefile() (translate.storage.csvl10n.csvfile class
parse_args() (trans-	method), 422
late.convert.convert.ArchiveConvertOptionParser	
	method), 428
method), 239	
parse_args() (trans-	parsefile() (translate.storage.html.htmlfile class
late.convert.convert.ConvertOptionParser	method), 437
method), 243	parsefile() (translate.storage.html.POHTMLParser
parse_args() (trans-	class method), 434
late.convert.po2moz.MozConvertOptionParser	parsefile() (translate.storage.ical.icalfile class
method), 254	method), 442
parse_args() (trans-	parsefile() (translate.storage.ini.inifile class
late.convert.po2tmx.TmxOptionParser	method), 447
method), 260	parsefile() (trans-
parse_args() (trans-	late.storage.jsonl10n.ARBJsonFile class
late.convert.po 2 word fast. Wf Option Parser	method), 453
method), 265	parsefile() (trans-
parse_args() (trans-	late.storage.jsonl10n.GoI18NJsonFile class

method), 458	parsefile() (trans-
parsefile() (translate.storage.jsonl10n.I18NextFile class method), 463	late.storage.properties.XWikiPageProperties class method), 595
parsefile() (translate.storage.jsonl10n.JsonFile class method), 468	parsefile() (translate.storage.pypo.pofile class method), 623
parsefile() (trans-	parsefile() (translate.storage.qm.qmfile class
late.storage.jsonl10n.JsonNestedFile class	method), 630
method), 470	<pre>parsefile() (translate.storage.qph.QphFile class</pre>
parsefile() (trans-	method), 635
late.storage.jsonl10n.WebExtensionJsonFile class method), 478	method), 641
parsefile() (translate.storage.lisa.LISAfile class	
method), 483	late.storage.subtitles.AdvSubStationAlphaFile
parsefile() (translate.storage.mo.mofile class	class method), 648
method), 489	parsefile() (trans-
parsefile() (trans-	late.storage.subtitles.MicroDVDFile class
late.storage.mozilla_lang.LangStore class	method), 649
method), 494	parsefile() (translate.storage.subtitles.SubRipFile
parsefile() (translate.storage.omegat.OmegaTFile	class method), 651
class method), 499	parsefile() (trans-
parsefile() (trans-	late.storage.subtitles.SubStationAlphaFile
late.storage.omegat.OmegaTFileTab class	class method), 653
method), 501	parsefile() (translate.storage.subtitles.SubtitleFile
parsefile() (translate.storage.php.LaravelPHPFile	class method), 655
class method), 559	parsefile() (translate.storage.tbx.tbxfile class
parsefile() (translate.storage.php.phpfile class	method), 660
method), 564	parsefile() (translate.storage.tiki.TikiStore class
parsefile() (translate.storage.pocommon.pofile class method), 570	method), 665 parsefile() (translate.storage.tmx.tmxfile class
parsefile() (translate.storage.poxliff.PoXliffFile	method), 671
class method), 578	parsefile() (trans-
parsefile() (translate.storage.properties.gwtfile	late.storage.trados.TradosTxtTmFile class
class method), 596	method), 680
	parsefile() (translate.storage.ts2.tsfile class
class method), 599	method), 682
parsefile() (trans-	parsefile() (translate.storage.txt.TxtFile class
late.storage.properties.javautf16file class	method), 687
method), 601	parsefile() (translate.storage.utx.UtxFile class
parsefile() (translate.storage.properties.javautf8file	method), 693
class method), 602	parsefile() (trans-
parsefile() (translate.storage.properties.joomlafile	late.storage.wordfast.WordfastTMFile class
class method), 604	method), 699
parsefile() (translate.storage.properties.propfile	<pre>parsefile() (translate.storage.xliff.xlifffile class</pre>
class method), 606	method), 706
<pre>parsefile() (translate.storage.properties.stringsfile</pre>	parseheader() (translate.storage.mo.mofile method),
class method), 614	489
parsefile() (trans-	parseheader() (translate.storage.pocommon.pofile
late.storage.properties.stringsutf8file class	method), 570
method), 616	parseheader() (trans-
parsefile() (translate.storage.properties.xwikifile	late.storage.poheader.poheader method),
class method), 617	575
parsefile() (trans-	parseheader() (translate.storage.poxliff.PoXliffFile
late.storage.properties.XWikiFullPage class	method), 578
method), 593	parseheader() (translate.storage.pypo.pofile

method), 624	method), 494
parseheaderstring() (in module trans- late.storage.poheader), 574	parsestring() (trans- late.storage.omegat.OmegaTFile class
ParseState (class in trans-	method), 500
late.storage.xml_extract.extract), 711	parsestring() (trans-
parseString() (in module translate.misc.ourdom), 398	late.storage.omegat.OmegaTFileTab class method), 501
parseString() (trans-	parsestring() (trans-
late.misc.ourdom.ExpatBuilderNS method), 397	late.storage.php.LaravelPHPFile class method), 559
parsestring() (translate.storage.base.DictStore class method), 404	parsestring() (translate.storage.php.phpfile class method), 564
parsestring() (trans-	<pre>parsestring() (translate.storage.pocommon.pofile</pre>
late.storage.base.TranslationStore class	class method), 570
method), 410	<pre>parsestring() (translate.storage.poxliff.PoXliffFile</pre>
parsestring() (trans-	class method), 578
	parsestring() (translate.storage.properties.gwtfile class method), 596
parsestring() (translate.storage.csvl10n.csvfile class method), 422	
parsestring() (translate.storage.dtd.dtdfile class	parsestring() (trans-
method), 428	late.storage.properties.javautf16file class
parsestring() (translate.storage.html.htmlfile class	method), 601
method), 437	parsestring() (trans-
parsestring() (trans-	late.storage.properties.javautf8file class
late.storage.html.POHTMLParser class	method), 602
method), 434	parsestring() (trans-
parsestring() (translate.storage.ical.icalfile class method), 442	late.storage.properties.joomlafile class method), 604
	parsestring() (translate.storage.properties.propfile class method), 606
parsestring() (trans-	parsestring() (trans-
late.storage.jsonl10n.ARBJsonFile class method), 453	late.storage.properties.stringsfile class method), 614
parsestring() (trans-	parsestring() (trans-
late.storage.jsonl10n.GoI18NJsonFile class method), 458	late.storage.properties.stringsutf8file class method), 616
	parsestring() (translate.storage.properties.xwikifile class method), 617
method), 463	parsestring() (trans-
parsestring() (translate.storage.jsonl10n.JsonFile class method), 468	late.storage.properties.XWikiFullPage class method), 593
parsestring() (trans-	parsestring() (trans-
late.storage.jsonl10n.JsonNestedFile class method), 470	late.storage.properties.XWikiPageProperties class method), 595
parsestring() (trans-	parsestring() (translate.storage.pypo.pofile class
late.storage.jsonl10n.WebExtensionJsonFile class method), 478	method), 624 parsestring() (translate.storage.qm.qmfile class
parsestring() (translate.storage.lisa.LISAfile class	method), 630
method), 483	parsestring() (translate.storage.qph.QphFile class
parsestring() (translate.storage.mo.mofile class	method), 635
method), 489	parsestring() (translate.storage.rc.rcfile class
parsestring() (trans-	method), 641
late.storage.mozilla_lang.LangStore class	parsestring() (trans-

late. storage. subtitles. Adv SubStation Alpha File	268
class method), 648	po2lang (class in translate.convert.po2mozlang), 252
parsestring() (trans-	po2tiki (class in translate.convert.po2tiki), 257
late.storage.subtitles.MicroDVDFile class	po2txt (class in translate.convert.po2txt), 262
method), 649	po2yaml (class in translate.convert.po2yaml), 266
parsestring() (trans-	pofile (class in translate.storage.pocommon), 568
late.storage.subtitles.SubRipFile class method),	pofile (class in translate.storage.pypo), 622
651	poheader (class in translate.storage.poheader), 574
parsestring() (trans-	POHTMLParser (class in translate.storage.html), 433
late.storage.subtitles.SubStationAlphaFile	pop () (translate.misc.dictutils.cidict method), 389
class method), 653	pop() (translate.storage.oo.unormalizechar method),
parsestring() (trans-	507
late.storage.subtitles.SubtitleFile class	<pre>popitem() (translate.misc.dictutils.cidict method), 389</pre>
method), 655	<pre>popitem() (translate.storage.oo.unormalizechar</pre>
<pre>parsestring() (translate.storage.tbx.tbxfile class</pre>	method), 507
method), 660	potifyformat() (trans-
<pre>parsestring() (translate.storage.tiki.TikiStore class</pre>	late.convert.convert.ArchiveConvertOptionParset method), 239
<pre>parsestring() (translate.storage.tmx.tmxfile class</pre>	potifyformat() (trans-
method), 671	late.convert.convert.ConvertOptionParser
parsestring() (trans-	method), 243
late.storage.trados.TradosTxtTmFile class	potifyformat() (trans-
method), 680	late.convert.po2moz.MozConvertOptionParser
parsestring() (translate.storage.ts2.tsfile class	method), 255
method), 682	potifyformat() (trans-
<pre>parsestring() (translate.storage.txt.TxtFile class</pre>	late.convert.po2tmx.TmxOptionParser method), 260
parsestring() (translate.storage.utx.UtxFile class	
method), 693	late.convert.po2wordfast.WfOptionParser
parsestring() (trans-	method), 265
late.storage.wordfast.WordfastTMFile class	pounit (class in translate.storage.pocommon), 571
method), 699	pounit (class in translate.storage.pypo), 625
parsestring() (translate.storage.xliff.xlifffile class	PoWrapper (class in translate.storage.pypo), 621
method), 706	PoXliffFile (class in translate.storage.poxliff), 576
<pre>partition() (translate.misc.multistring.multistring</pre>	PoXliffUnit (class in translate.storage.poxliff), 579
method), 392	<pre>pretranslate_file() (in module trans-</pre>
Ph (class in translate.storage.placeables.base), 511	late.tools.pretranslate), 729
Ph (class in translate.storage.placeables.xliff), 554	<pre>pretranslate_store() (in module trans-</pre>
php2po (class in translate.convert.php2po), 250	late.tools.pretranslate), 730
phpdecode() (in module translate.storage.php), 563	<pre>pretranslate_unit() (in module trans-</pre>
phpencode () (in module translate.storage.php), 563	late.tools.pretranslate), 730
phpfile (class in translate.storage.php), 563	<pre>prev_source (translate.storage.pypo.pounit at-</pre>
phpunit (class in translate.storage.php), 565	tribute), 627
pluralequation (translate.lang.common.Common	<pre>print_help() (trans-</pre>
attribute), 363	late.convert.convert.ArchiveConvertOptionParser
po2dtd (class in translate.convert.po2dtd), 250	method), 239
po2html (class in translate.convert.po2html), 251	<pre>print_help() (trans-</pre>
po2ical (class in translate.convert.po2ical), 251	late.convert.convert.ConvertOptionParser
po2inc() (in module trans-	method), 243
late.convert.prop2mozfunny), 267	<pre>print_help() (trans-</pre>
po2ini (class in translate.convert.po2ini), 251	late.convert.po2moz.MozConvertOptionParser
po2ini() (in module trans-	method), 255
late.convert.prop2mozfunny), 267	<pre>print_help() (trans-</pre>
po2it() (in module translate.convert.prop2mozfunny),	late.convert.po2tmx.TmxOptionParser

method), 260		method), 728
<pre>print_help() late.convert.po2wordfast.WfOptionPar</pre>	(trans-	<pre>print_tree() (translate.storage.placeables.base.Bpt method), 509</pre>
method), 265	1361	print_tree() (translate.storage.placeables.base.Bx
print_help()	(trans-	method), 518
late.filters.pofilter.FilterOptionParser	`	print_tree() (translate.storage.placeables.base.Ept
354	memou),	method), 511
print_help()	(trans-	<pre>print_tree() (translate.storage.placeables.base.Ex</pre>
late.misc.optrecurse.RecursiveOption I	Parser	method), 519
method), 395		<pre>print_tree() (translate.storage.placeables.base.G</pre>
<pre>print_help()</pre>	(trans-	method), 516
late.tools.poconflicts.ConflictOptionPo method), 717	arser	print_tree() (translate.storage.placeables.base.It method), 514
print_help()	(trans-	<pre>print_tree() (translate.storage.placeables.base.Ph</pre>
	method),	method), 513
721	,,	<pre>print_tree() (translate.storage.placeables.base.Sub</pre>
print_help()	(trans-	method), 522
late.tools.porestructure.SplitOptionPa method), 724	rser	<pre>print_tree() (translate.storage.placeables.base.X method), 521</pre>
print_help()	(trans-	print_tree() (trans-
late.tools.poterminology.Terminology(
method), 728	1	method), 524
print_manpage()	(trans-	<pre>print_tree()</pre> (trans-
late.convert.convert.ArchiveConvertOpmethod), 239		
print_manpage()	(trans-	print_tree() (trans-
late.convert.convert.ConvertOptionPa. method), 243		late.storage.placeables.general.XMLTagPlaceable method), 528
print_manpage()	(trans-	print_tree() (trans-
late.convert.po2moz.MozConvertOptic method), 255		late.storage.placeables.interfaces.BasePlaceable method), 529
print_manpage()	(trans-	print_tree() (trans-
late.convert.po2tmx.TmxOptionParser		late.storage.placeables.interfaces.InvisiblePlaceable
method), 260		method), 531
print_manpage()	(trans-	<pre>print_tree() (trans-</pre>
late.convert.po2wordfast.WfOptionParmethod), 265	rser	late.storage.placeables.interfaces.MaskingPlaceable method), 533
print_manpage()	(trans-	<pre>print_tree()</pre> (trans-
late.filters.pofilter.FilterOptionParser 354		late.storage.placeables.interfaces.ReplacementPlaceable method), 534
print_manpage()	(trans-	print_tree() (trans-
late.misc.optrecurse.RecursiveOption1 method), 395	Parser	late.storage.placeables.interfaces.SubflowPlaceable method), 536
print_manpage()	(trans-	print_tree() (trans-
late.tools.poconflicts.ConflictOptionPa	`	late.storage.placeables.strelem.StringElem
method), 717		method), 539
print_manpage()	(trans-	print_tree() (trans-
late.tools.pogrep.GrepOptionParser	method),	late. storage. placeables. terminology. Terminology Placeables and the property of the prope
721		method), 541
print_manpage()		print_tree() (translate.storage.placeables.xliff.Bpt
late.tools.porestructure.SplitOptionPa	rser	method), 542
method), 724	(+	print_tree() (translate.storage.placeables.xliff.Bx
print_manpage()	(trans-	method), 547
tate.toots.poterminotogy.1erminotogy(ριιοnPars	eprint_tree() (translate.storage.placeables.xliff.Ept

method), 544	method), 243
<pre>print_tree() (translate.storage.placeables.xliff.Ex</pre>	<pre>print_version() (trans-</pre>
method), 549	late.convert.po2moz.MozConvertOptionParser
$print_tree()$ (translate.storage.placeables.xliff.G	method), 255
method), 550	<pre>print_version() (trans-</pre>
<pre>print_tree() (translate.storage.placeables.xliff.It</pre>	late.convert.po2tmx.TmxOptionParser method), 260
<pre>print_tree() (translate.storage.placeables.xliff.Ph</pre>	<pre>print_version() (trans-</pre>
<pre>method), 555 print_tree() (translate.storage.placeables.xliff.Sub</pre>	late.convert.po2wordfast.WfOptionParser method), 265
method), 554	<pre>print_version() (trans-</pre>
<pre>print_tree() (trans-</pre>	late.filters.pofilter.FilterOptionParser method),
late.storage.placeables.xliff.UnknownXML method), 557	354 print_version() (trans-
<pre>print_tree() (translate.storage.placeables.xliff.X</pre>	late.misc.optrecurse.RecursiveOptionParser
method), 546	method), 395
	<pre>print_version() (trans-</pre>
late.convert.convert.ArchiveConvertOptionParse method), 239	r late.tools.poconflicts.ConflictOptionParser method), 717
	<pre>print_version() (trans-</pre>
late.convert.convert.ConvertOptionParser method), 243	late.tools.pogrep.GrepOptionParser method), 721
<pre>print_usage() (trans-</pre>	<pre>print_version() (trans-</pre>
late.convert.po2moz.MozConvertOptionParser method), 255	late.tools.porestructure.SplitOptionParser method), 724
<pre>print_usage() (trans-</pre>	<pre>print_version() (trans-</pre>
late.convert.po2tmx.TmxOptionParser method), 260	late.tools.poterminology.TerminologyOptionParser method), 728
<pre>print_usage() (trans-</pre>	<pre>printf() (translate.filters.checks.CCLicenseChecker</pre>
late. convert. po 2 word fast. Wf Option Parser	method), 276
method), 265	<pre>printf() (translate.filters.checks.DrupalChecker</pre>
print_usage() (trans-	method), 282
late.filters.pofilter.FilterOptionParser method), 354	printf() (translate.filters.checks.GnomeChecker method), 288
	printf() (translate.filters.checks.IOSChecker
late.misc.optrecurse.RecursiveOptionParser	method), 293
method), 395	printf() (translate.filters.checks.KdeChecker
print_usage() (trans-	
late.tools.poconflicts.ConflictOptionParser method), 717	printf() (translate.filters.checks.L20nChecker method), 305
print_usage() (trans-	printf() (translate.filters.checks.LibreOfficeChecker
late.tools.pogrep.GrepOptionParser method), 721	method), 311 printf() (translate.filters.checks.MinimalChecker
print_usage() (trans-	method), 316
late.tools.porestructure.SplitOptionParser method), 724	<pre>printf() (translate.filters.checks.MozillaChecker</pre>
<pre>print_usage() (trans-</pre>	<pre>printf() (translate.filters.checks.OpenOfficeChecker</pre>
late.tools.poterminology.TerminologyOptionPars	
method), 728	printf() (translate.filters.checks.ReducedChecker
print_version() (trans-	method), 334
late.convert.convert.ArchiveConvertOptionParse method), 239	method), 339
<pre>print_version() (trans-</pre>	printf() (translate.filters.checks.TermChecker
late.convert.convert.ConvertOptionParser	method), 346

```
process_translatable() (in module
                                                                           trans-
                                                                                                       method), 518
              late.storage.xml_extract.extract), 711
                                                                                                                     (translate.storage.placeables.base.Ept
                                                                                        prune()
processfile()
                                                                          (trans-
                                                                                                      method), 511
              late.convert.convert.ArchiveConvertOptionParser prune ()
                                                                                                                      (translate.storage.placeables.base.Ex
              method), 240
                                                                                                       method), 519
processfile()
                                                                                        prune () (translate.storage.placeables.base.G method),
                                                                          (trans-
              late.convert.convert.ConvertOptionParser
              method), 243
                                                                                        prune () (translate.storage.placeables.base.It method),
processfile()
                                                                          (trans-
                                                                                                       514
              late.convert.po 2 moz. Moz Convert Option Parser \\
                                                                                                                     (translate.storage.placeables.base.Ph
                                                                                        prune()
              method), 255
                                                                                                       method), 513
processfile()
                                                                                                                    (translate.storage.placeables.base.Sub
                                                                          (trans-
                                                                                        prune()
              late.convert.po2tmx.TmxOptionParser
                                                                                                       method), 522
              method), 260
                                                                                        prune () (translate.storage.placeables.base.X method),
processfile()
                                                                          (trans-
              late.convert.po2wordfast.WfOptionParser
                                                                                        prune() (translate.storage.placeables.general.AltAttrPlaceable
              method), 265
                                                                                                       method), 524
processfile()
                                                                          (trans-
                                                                                        prune () (translate.storage.placeables.general.XMLEntityPlaceable
              late.filters.pofilter.FilterOptionParser method),
                                                                                                       method), 526
                                                                                        prune () (translate.storage.placeables.general.XMLTagPlaceable
                                                                          (trans-
processfile()
                                                                                                       method), 528
              late.misc.optrecurse.RecursiveOptionParser
                                                                                        prune() (translate.storage.placeables.interfaces.BasePlaceable
              method), 395
                                                                                                       method), 529
processfile()
                                                                                        prune () (translate.storage.placeables.interfaces.InvisiblePlaceable
                                                                          (trans-
              late.tools.poconflicts.ConflictOptionParser
                                                                                                       method), 531
                                                                                        \verb"prune" () (translate.storage.placeables.interfaces.Masking Placeable
              method), 717
processfile()
                                                                          (trans-
                                                                                                       method), 533
              late.tools.pogrep.GrepOptionParser
                                                                                        prune() (translate.storage.placeables.interfaces.ReplacementPlaceable
                                                                      method),
                                                                                                       method), 534
processfile()
                                                                          (trans-
                                                                                        prune() (translate.storage.placeables.interfaces.SubflowPlaceable
              late.tools.porestructure.SplitOptionParser
                                                                                                       method), 536
              method), 724
                                                                                        prune() (translate.storage.placeables.strelem.StringElem
processfile()
                                                                          (trans-
                                                                                                       method), 539
              late. tools. poterminology. Terminology Option Parse \verb|prune|| () (translate. storage. placeables. terminology. Terminology Placeables () (translate. storage. placeables () (trans
              method), 728
                                                                                                       method), 541
ProgressBar (class in translate.misc.progressbar),
                                                                                                                     (translate.storage.placeables.xliff.Bpt
                                                                                        prune()
              398
                                                                                                       method), 542
Project (class in translate.storage.project), 583
                                                                                        prune() (translate.storage.placeables.xliff.Bx method),
ProjectStore (class in translate.storage.projstore),
                                                                                                       547
              584
                                                                                                                     (translate.storage.placeables.xliff.Ept
                                                                                        prune()
prop2inc()
                                                    module
                                   (in
                                                                           trans-
                                                                                                      method), 544
                                                                                        prune() (translate.storage.placeables.xliff.Ex method),
              late.convert.prop2mozfunny), 268
prop2it()
                                 (in
                                                   module
                                                                            trans-
              late.convert.prop2mozfunny), 268
                                                                                        prune() (translate.storage.placeables.xliff.G method),
prop2po (class in translate.convert.prop2po), 268
propertiesdecode()
                                                          module
                                                                                        prune() (translate.storage.placeables.xliff.It method),
                                                                            trans-
              late.misc.quote), 399
propfile (class in translate.storage.properties), 605
                                                                                        prune () (translate.storage.placeables.xliff.Ph method),
proppluralunit
                                          (class
                                                             in
                                                                            trans-
                                                                                                       555
                                                                                                                     (translate.storage.placeables.xliff.Sub
              late.storage.properties), 607
                                                                                        prune()
propunit (class in translate.storage.properties), 609
                                                                                                       method), 554
                                                                                        prune() (translate.storage.placeables.xliff.UnknownXML
                            (translate.storage.placeables.base.Bpt
              method), 510
                                                                                                       method), 557
                             (translate.storage.placeables.base.Bx prune() (translate.storage.placeables.xliff.X method),
prune()
```

546	method), 366
puncdict (translate.lang.common.Common attribute), 363	punctranslate() (translate.lang.el.el class method), 367
puncend() (in module translate.filters.decoration), 352 puncspacing() (trans-	punctranslate() (translate.lang.es.es class method), 367
late.filters.checks.CCLicenseChecker method), 276	punctranslate() (translate.lang.fa.fa class method), 368
puncspacing() (trans- late.filters.checks.DrupalChecker method),	punctranslate() (translate.lang.fi.fi class method), 369
282 puncspacing() (trans-	punctranslate() (translate.lang.fr.fr class method), 370
late.filters.checks.GnomeChecker method), 288	punctranslate() (translate.lang.gu.gu class method), 371
puncspacing() (translate.filters.checks.IOSChecker method), 294	punctranslate() (translate.lang.he.he class method), 371
puncspacing() (translate.filters.checks.KdeChecker method), 299	punctranslate() (translate.lang.hi.hi class method), 372
puncspacing() (translate.filters.checks.L20nChecker method), 305	punctranslate() (translate.lang.hy.hy class method), 373
puncspacing() (trans- late.filters.checks.LibreOfficeChecker method),	punctranslate() (translate.lang.ja.ja class method), 374
311	punctranslate() (translate.lang.km.km class
puncspacing() (trans- late.filters.checks.MinimalChecker method), 317	method), 375 punctranslate() (translate.lang.kn.kn class method), 375
puncspacing() (trans- late.filters.checks.MozillaChecker method),	punctranslate() (translate.lang.ko.ko class method), 376
322 puncspacing() (trans-	punctranslate() (translate.lang.ml.ml class
puncspacing() (trans- late.filters.checks.OpenOfficeChecker method), 328	method), 377 punctranslate() (translate.lang.mr.mr class method), 378
puncspacing() (trans-	punctranslate() (translate.lang.ne.ne class
late.filters.checks.ReducedChecker method), 334	method), 379 punctranslate() (translate.lang.pa.pa class
puncspacing() (trans- late.filters.checks.StandardChecker method),	method), 379 punctranslate() (translate.lang.si.si class method),
340	381
puncspacing() (translate.filters.checks.TermChecker method), 347	382
puncstart () (in module translate.filters.decoration), 352	punctranslate() (translate.lang.sv.sv class method), 382
<pre>punctranslate() (translate.lang.af.af class method), 357</pre>	punctranslate() (translate.lang.ta.ta class method), 383
punctranslate() (translate.lang.am.am class method), 358	<pre>punctranslate() (translate.lang.te.te class method),</pre>
punctranslate() (translate.lang.ar.ar class method), 359	punctranslate() (translate.lang.th.th class method), 385
punctranslate() (translate.lang.bn.bn class method), 360	punctranslate() (translate.lang.ug.ug class method), 386
<pre>punctranslate() (translate.lang.code_or.code_or</pre>	punctranslate() (translate.lang.ur.ur class method), 386
punctranslate() (translate.lang.common.Common class method), 363	punctranslate() (translate.lang.vi.vi class method), 387
punctranslate() (translate.lang.de.de class	punctranslate() (translate.lang.zh.zh class

method), 388	<pre>pythonbraceformat() (trans-</pre>
punctuation (translate.lang.common.Common attribute), 363	late.filters.checks.MinimalChecker method), 317
purepunc () (translate.filters.checks.CCLicenseChecker method), 276	pythonbraceformat() (trans- late.filters.checks.MozillaChecker method),
purepunc() (translate.filters.checks.DrupalChecker method), 282	323 pythonbraceformat() (trans-
purepunc() (translate.filters.checks.GnomeChecker method), 288	late.filters.checks.OpenOfficeChecker method), 328
purepunc() (translate.filters.checks.IOSChecker method), 294	$\begin{tabular}{ll} pythonbraceformat() & (trans-\\ late.filters.checks.ReducedChecker & method), \end{tabular}$
purepunc() (translate.filters.checks.KdeChecker	334
method), 299 purepunc () (translate.filters.checks.L20nChecker method), 305	pythonbraceformat() (trans- late.filters.checks.StandardChecker method), 340
purepunc() (translate.filters.checks.LibreOfficeChecker	
method), 311 purepunc() (translate.filters.checks.MinimalChecker	late.filters.checks.TermChecker method), 347
method), 317 purepunc() (translate.filters.checks.MozillaChecker	Q
method), 323	qmfile (class in translate.storage.qm), 629
<pre>purepunc() (translate.filters.checks.OpenOfficeChecker</pre>	qmunit (class in translate.storage.qm), 630 qmunpack() (in module translate.storage.qm), 633
purepunc() (translate.filters.checks.ReducedChecker method), 334	QphFile (class in translate.storage.qph), 634 QphUnit (class in translate.storage.qph), 636
purepunc() (translate.filters.checks.StandardChecker method), 340	<pre>quote_plus() (in module trans- late.storage.pocommon), 573</pre>
purepunc() (translate.filters.checks.TermChecker method), 347	<pre>quoteforandroid() (in module trans- late.storage.dtd), 432</pre>
Python Enhancement Proposals PEP 257,169	<pre>quotefordtd() (in module translate.storage.dtd),</pre>
PEP 8, 161, 163, 166 python_distance() (in module trans-	quoteforpo() (in module translate.storage.pypo), 628
<pre>late.search.lshtein), 401 pythonbraceformat() (trans-</pre>	quotes (translate.lang.common.Common attribute), 363
late.filters.checks.CCLicenseChecker method), 276	R
pythonbraceformat() (trans- late.filters.checks.DrupalChecker method), 282	<pre>rc2po (class in translate.convert.rc2po), 269 rc_statement() (in module translate.storage.rc), 640</pre>
pythonbraceformat() (trans- late.filters.checks.GnomeChecker method),	rcfile (class in translate.storage.rc), 640 rcunit (class in translate.storage.rc), 642
288 pythonbraceformat() (trans-	read_obsolete_lines() (in module trans- late.storage.poparser), 575
late.filters.checks.IOSChecker method), 294 pythonbraceformat() (trans-	read_prevmsgid_lines() (in module trans-
late.filters.checks.KdeChecker method),	<pre>late.storage.poparser), 575 real_index() (in module translate.tools.pogrep), 722 reclassifyunit() (trans-</pre>
pythonbraceformat() (trans-	late.storage.statistics Statistics method),
late.filters.checks.L20nChecker method),	645
305 pythonbraceformat() (trans-	recursearchivefiles() (trans-
late.filters.checks.LibreOfficeChecker method), 311	late.convert.convert.ArchiveConvertOptionParset method), 240

late.convert.po2tmx.TmxOptionParser	recurseinputfiles() (trans- late.filters.pofilter.FilterOptionParser method), 354
method), 260 recursearchivefiles() (trans-	recurseinputfiles() (trans-
late.convert.po2wordfast.WfOptionParser method), 265	late.misc.optrecurse.RecursiveOptionParser method), 396
recurseinputfilelist() (trans- late.convert.convert.ArchiveConvertOptionParse method), 240	recurseinputfiles() (trans- r late.tools.poconflicts.ConflictOptionParser method), 717
recurseinputfilelist() (trans-	recurseinputfiles() (trans-
late.convert.convert.ConvertOptionParser method), 243	late.tools.pogrep.GrepOptionParser method), 721
recurseinputfilelist() (trans- late.convert.po2moz.MozConvertOptionParser method), 255	recurseinputfiles() (trans- late.tools.porestructure.SplitOptionParser method), 725
	recurseinputfiles() (trans-
late.convert.po2tmx.TmxOptionParser method), 260	late.tools.poterminology.TerminologyOptionParser method), 728
recurseinputfilelist() (trans- late.convert.po2wordfast.WfOptionParser	RecursiveOptionParser (class in trans- late.misc.optrecurse), 394
method), 265	recursiveprocess() (trans-
recurseinputfilelist() (trans- late.filters.pofilter.FilterOptionParser method),	late.convert.convert.ArchiveConvertOptionParser method), 240
354	recursiveprocess() (trans-
recurseinputfilelist() (trans-	late.convert.convert.ConvertOptionParser
late.misc.optrecurse.RecursiveOptionParser	method), 243
method), 396	recursiveprocess() (trans-
recurseinputfilelist() (trans- late.tools.poconflicts.ConflictOptionParser	late.convert.po2moz.MozConvertOptionParser method), 255
method), 717	recursiveprocess() (trans-
recurseinputfilelist() (trans-	late.convert.po2tmx.TmxOptionParser
late.tools.pogrep.GrepOptionParser method),	method), 261
721	recursiveprocess() (trans-
recurseinputfilelist() (trans- late.tools.porestructure.SplitOptionParser	late.convert.po2wordfast.WfOptionParser method), 265
method), 724	recursiveprocess() (trans-
recurseinputfilelist() (trans-	$late. filters. pofilter. Filter Option Parser\ method),$
late.tools.poterminology.TerminologyOptionPars	
method), 728	recursiveprocess() (trans-
recurseinputfiles() (trans-	late.misc.optrecurse.RecursiveOptionParser
late.convert.convert.ArchiveConvertOptionParse method), 240	r method), 396 recursiveprocess() (trans-
recurseinputfiles() (trans-	late.tools.poconflicts.ConflictOptionParser
late.convert.convert.ConvertOptionParser method), 243	method), 717 recursiveprocess() (trans-
recurseinputfiles() (trans-	late.tools.pogrep.GrepOptionParser method),
late.convert.po2moz.MozConvertOptionParser	721
method), 255	recursiveprocess() (trans-
recurseinputfiles() (trans-	late.tools.porestructure.SplitOptionParser
late.convert.po2tmx.TmxOptionParser	method), 725
method), 260	recursiveprocess() (trans-
recurseinputfiles() (trans-	late.tools.poterminology.TerminologyOptionParser
late.convert.po2wordfast.WfOptionParser	method), 728
method), 265	redtd (class in translate.convert.po2dtd), 250

reduce_	_tree() (in module	trans-	remove	_type()	(trans-
	late.storage.xml_extract.misc), 712			late.storage.placeables.interfaces.Inv	isiblePlaceable
Reduce	dChecker (class in translate.filter	s.checks),		method), 531	
	331		remove	_type()	(trans-
regist	er_dialect() (in module	trans-		late.storage.placeables.interfaces.Ma	skingPlaceable
	late.storage.ini), 451			method), 533	
regist	er_dialect() (in module	trans-	remove	_type()	(trans-
_	late.storage.properties), 613			late.storage.placeables.interfaces.Rep	placementPlaceable
reinde	nt () (in module translate.misc.xml	_helpers),		method), 534	
	400	•	remove	_type()	(trans-
remove	_file()	(trans-		late.storage.placeables.interfaces.Sul	bflowPlaceable
	late.storage.bundleprojstore.BundlePr	rojectStore		method), 536	
	method), 414		remove	_type()	(trans-
remove	_file() (translate.storage.proje	ct.Project		late.storage.placeables.strelem.String	Elem
	method), 584			method), 539	
remove	_file()	(trans-	remove	_type()	(trans-
	late.storage.projstore.ProjectStore	method),		late.storage.placeables.terminology.T	TerminologyPlaceabl
	584			method), 541	
remove	_spreadsheet_escapes()	(trans-	remove	_type()	(trans-
	late.storage.csvl10n.csvunit method),	425		late.storage.placeables.xliff.Bpt	method),
remove	_type()	(trans-		542	
	late.storage.placeables.base.Bpt	method),	remove	_type() (translate.storage.placeab	les.xliff.Bx
	510			method), 547	
remove	_type() (translate.storage.placeable	es.base.Bx	remove	_type()	(trans-
	method), 518			late.storage.placeables.xliff.Ept	method),
remove	_type()	(trans-		544	
	late.storage.placeables.base.Ept	method),	remove	_type() (translate.storage.placeab	les.xliff.Ex
	511			method), 549	
remove	_type() (translate.storage.placeable	es.base.Ex	remove	_type() (translate.storage.placeal	bles.xliff.G
	method), 519			<i>method</i>), 551	
remove	_type() (translate.storage.placeabl	es.base.G	remove	_type() (translate.storage.placea	bles.xliff.It
	method), 516			method), 552	
remove_	_type() (translate.storage.placeab	les.base.It	remove	_type() (translate.storage.placeab	les.xliff.Ph
	<i>method</i>), 514			method), 555	
remove	_type()	(trans-	remove	_type()	(trans-
	late.storage.placeables.base.Ph	method),		late.storage.placeables.xliff.Sub	method),
	513			554	
remove_	_type()	(trans-	remove	_type()	(trans-
	late.storage.placeables.base.Sub	method),		late.storage.placeables.xliff.Unknown	iXML
	522			method), 557	
remove_	_type() (translate.storage.placeab	les.base.X	remove	_type() (translate.storage.placea	bles.xliff.X
	<i>method</i>), 521			method), 546	
remove_	_type()	(trans-	remove	_unit_from_index()	(trans-
	late. storage. placeables. general. Alt Att	rPlaceable		$late. storage. base. Dict Store\ method),$	404
	method), 524		remove	_unit_from_index()	(trans-
remove_	_type()	(trans-		late.storage.base.TranslationStore	method),
	late.storage.placeables.general.XMLE	EntityPlacea		410	
	method), 526		remove	_unit_from_index()	(trans-
remove_	_type()	(trans-		late.storage.catkeys.CatkeysFile	method),
	late.storage.placeables.general.XMLT	TagPlaceabl		416	
	method), 528		remove	_unit_from_index()	(trans-
remove_	_type()	(trans-		late.storage.csvl10n.csvfile method),	
	late.storage.placeables.interfaces.Bas	ePlaceable	remove		(trans-
	method), 529			late.storage.dtd.dtdfile method), 428	

remove_	_unit_from_index() late.storage.html.htmlfile method), 43	(trans-	late.storage.properties.javautf16file 601	method),
remove_	_unit_from_index() late.storage.html.POHTMLParser 434	(trans- method),	remove_unit_from_index() late.storage.properties.javautf8file 602	(trans- method),
remove_	_unit_from_index() late.storage.ical.icalfile method), 442	(trans-	remove_unit_from_index() late.storage.properties.joomlafile	(trans- method),
remove_	_unit_from_index() late.storage.ini.inifile method), 447	(trans-	604 remove_unit_from_index()	(trans-
remove_	_unit_from_index() late.storage.jsonl10n.ARBJsonFile	(trans- method),	late.storage.properties.propfile 606	method),
remove_	453 _unit_from_index() late.storage.jsonl10n.GoI18NJsonFile method), 458	(trans-	remove_unit_from_index() late.storage.properties.stringsfile 614 remove_unit_from_index()	(trans- method), (trans-
remove_	_unit_from_index() late.storage.jsonl10n.I18NextFile	(trans- method),	late.storage.properties.stringsutf8file 616	method),
remove_	463 _unit_from_index() late.storage.jsonl10n.JsonFile method	(trans-	remove_unit_from_index() late.storage.properties.xwikifile 617	(trans- method),
remove_	_unit_from_index() late.storage.jsonl10n.JsonNestedFile 470	(trans-	remove_unit_from_index() late.storage.properties.XWikiFullPage method), 593	(trans-
remove_	_unit_from_index() late.storage.jsonl10n.WebExtensionJs method), 478	(trans- onFile	remove_unit_from_index() late.storage.properties.XWikiPageProp method), 595	(trans- perties
remove_	_unit_from_index() late.storage.lisa.LISAfile method), 483	(trans-	<pre>remove_unit_from_index() late.storage.pypo.pofile method), 624</pre>	(trans-
remove_	_unit_from_index() late.storage.mo.mofile method), 489	(trans-	remove_unit_from_index() late.storage.qm.qmfile method), 630	(trans-
remove_	_unit_from_index() late.storage.mozilla_lang.LangStore 494	(trans- method),	<pre>remove_unit_from_index() late.storage.qph.QphFile method), 635 remove_unit_from_index()</pre>	(trans- (trans-
remove_	_unit_from_index() late.storage.omegat.OmegaTFile 500	(trans- method),	<pre>late.storage.rc.rcfile method), 641 remove_unit_from_index() late.storage.subtitles.AdvSubStationAl</pre>	(trans- lphaFile
	_unit_from_index() late.storage.omegat.OmegaTFileTab 501	(trans- method),	method), 648 remove_unit_from_index() late.storage.subtitles.MicroDVDFile	(trans- method),
remove_	_unit_from_index() late.storage.php.LaravelPHPFile 559	(trans- method),	remove_unit_from_index() late.storage.subtitles.SubRipFile	(trans-method),
remove_	_unit_from_index() late.storage.php.phpfile method), 564	(trans-	651 remove_unit_from_index()	(trans-
remove_	_unit_from_index() late.storage.pocommon.pofile method	(trans-	late.storage.subtitles.SubStationAlpha method), 653	
	_unit_from_index() late.storage.poxliff.PoXliffFile method _unit_from_index()	(trans- 2), 578 (trans-	remove_unit_from_index() late.storage.subtitles.SubtitleFile 655	(trans- method),
	late.storage.properties.gwtfile method _unit_from_index()		remove_unit_from_index() late.storage.tbx.tbxfile method), 660	(trans-
Temove	_unit_from_index() late.storage.properties.javafile 599	method),	remove_unit_from_index() late.storage.tiki.TikiStore method), 660	(trans-
remove_	_unit_from_index()	(trans-	remove_unit_from_index()	(trans-

late.storage.tmx.tmxfile method), 671	473
<pre>remove_unit_from_index() (trans-</pre>	removenotes() (translate.storage.jsonl10n.JsonUnit
late.storage.trados.TradosTxtTmFile method),	method), 476
680	removenotes() (trans-
<pre>remove_unit_from_index() (trans-</pre>	late.storage.jsonl10n.WebExtensionJsonUnit
late.storage.ts2.tsfile method), 682	method), 481
remove_unit_from_index() (trans-	removenotes() (translate.storage.lisa.LISAunit
late.storage.txt.TxtFile method), 688	method), 486
remove_unit_from_index() (trans-	removenotes() (translate.storage.mo.mounii
late.storage.utx.UtxFile method), 693	method), 492
remove_unit_from_index() (trans-	removenotes() (trans-
late.storage.wordfast.WordfastTMFile method),	late.storage.mozilla_lang.LangUnit method),
699	497
	removenotes() (trans-
late.storage.xliff.xlifffile method), 706	late.storage.omegat.OmegaTUnit method),
removedefaultfile() (trans-	505
late.storage.poxliff.PoXliffFile method), 578	removenotes() (trans-
removedefaultfile() (trans-	late.storage.php.LaravelPHPUnit method),
late.storage.xliff.xlifffile method), 706	562
removeduplicates() (translate.storage.pypo.pofile	removenotes() (translate.storage.php.phpunit
method), 624	method), 567
removeinvalidamps() (in module trans-	removenotes() (translate.storage.pocommon.pounit
late.storage.dtd), 432	<i>method</i>), 573
removekdecomments() (in module trans-	removenotes() (translate.storage.poxliff.PoXliffUnit
late.filters.prefilters), 356	method), 582
removenotes() (translate.storage.base.DictUnit	removenotes() (trans-
method), 408	late.storage.properties.proppluralunit method),
removenotes() (trans-	609
late.storage.base.TranslationUnit method),	removenotes() (trans-
413	late.storage.properties.propunit method),
removenotes() (trans-	612
late.storage.catkeys.CatkeysUnit method),	removenotes() (trans-
420	late.storage.properties.xwikiunit method),
removenotes() (translate.storage.csvl10n.csvunit	620
method), 425	
	removenotes() (translate.storage.pypo.pounit method), 627
removenotes () (translate.storage.dtd.dtdunit	
method), 431	removenotes() (translate.storage.qm.qmunit
removenotes() (translate.storage.html.htmlunit	
method), 440	removenotes() (translate.storage.qph.QphUnit
removenotes() (translate.storage.ical.icalunit	method), 638
method), 445	removenotes() (translate.storage.rc.rcunit method)
removenotes () (translate.storage.ini.iniunit method),	644
450	removenotes() (trans-
removenotes() (trans-	late.storage.subtitles.SubtitleUnit method),
late.storage.jsonl10n.ARBJsonUnit method),	658
456	removenotes() (translate.storage.tbx.tbxunit
removenotes() (trans-	method), 663
late.storage.jsonl10n.GoI18NJsonUnit	removenotes() (translate.storage.tiki.TikiUnit
method), 461	method), 668
removenotes() (trans-	removenotes() (translate.storage.tmx.tmxunii
late.storage.jsonl10n.I18NextUnit method),	method), 674
466	removenotes() (translate.storage.trados.TradosUnit
removenotes() (trans-	method), 678
late.storage.jsonl10n.JsonNestedUnit method),	removenotes() (translate.storage.ts2.tsunit method).
inic.siorage.jsomion.ssomvesieuonii memou),	1 cmo v cmo cos () (nansime.sionage.is2.ismin memou),

685	removeunit() (translate.storage.omegat.OmegaTFile
removenotes () (translate.storage.txt.TxtUnit	method), 500
method), 691	removeunit() (trans-
removenotes () (translate.storage.utx.UtxUnit method), 696	late.storage.omegat.OmegaTFileTab method), 501
removenotes() (trans-	removeunit() (trans-
late.storage.wordfast.WordfastUnit method), 703	late.storage.php.LaravelPHPFile method), 559
removenotes () (translate.storage.xliff.xliffunit method), 710	removeunit() (translate.storage.php.phpfile method), 564
removeunit() (translate.storage.base.DictStore method), 404	removeunit() (translate.storage.pocommon.pofile method), 570
removeunit() (trans-	removeunit() (translate.storage.poxliff.PoXliffFile
late.storage.base.TranslationStore method), 410	<pre>method), 578 removeunit() (translate.storage.properties.gwtfile</pre>
removeunit() (translate.storage.catkeys.CatkeysFile	method), 596
method), 416	removeunit() (translate.storage.properties.javafile
removeunit() (translate.storage.csvl10n.csvfile	method), 599
method), 422	removeunit() (trans-
removeunit () (translate.storage.dtd.dtdfile method), 428	late.storage.properties.javautf16file method), 601
removeunit() (translate.storage.html.htmlfile	removeunit() (trans-
<pre>method), 437 removeunit() (trans-</pre>	late.storage.properties.javautf8file method), 602
late.storage.html.POHTMLParser method),	removeunit() (trans-
435	late.storage.properties.joomlafile method),
removeunit()(translate.storage.ical.icalfile method),	604
442	removeunit() (translate.storage.properties.propfile
<pre>removeunit() (translate.storage.ini.inifile method),</pre>	method), 606
447	removeunit () (translate.storage.properties.stringsfile
removeunit() (trans-	method), 614
late.storage.jsonl10n.ARBJsonFile method),	removeunit() (trans-
453	late.storage.properties.stringsutf8file method),
removeunit() (trans-	616
late.storage.jsonl10n.GoI18NJsonFile method), 458	removeunit() (translate.storage.properties.xwikifile method), 617
removeunit() (trans-	removeunit() (trans-
late.storage.jsonl10n.I18NextFile method), 463	late.storage.properties.XWikiFullPage method), 593
$\verb"removeunit"() \qquad \textit{(translate.storage.jsonl10n.JsonFile}$	removeunit() (trans-
<pre>method), 468 removeunit() (trans-</pre>	late.storage.properties.XWikiPageProperties method), 595
late.storage.jsonl10n.JsonNestedFile method), 470	removeunit() (translate.storage.pypo.pofile method), 624
removeunit() (trans-	removeunit() (translate.storage.qm.qmfile method),
late.storage.jsonl10n.WebExtensionJsonFile	630
method), 478	removeunit() (translate.storage.qph.QphFile
removeunit() (translate.storage.lisa.LISAfile	method), 635
method), 483	removeunit() (translate.storage.rc.rcfile method),
<pre>removeunit() (translate.storage.mo.mofile method),</pre>	641
489	removeunit() (trans-
removeunit() (trans-	
	late.storage.subtitles.AdvSubStationAlphaFile
late.storage.mozilla_lang.LangStore method), 494	late.storage.subtitles.AdvSubStationAlphaFile method), 648 removeunit() (trans-

late.storage.subtitles.MicroDVDFile method), 649	<pre>require_index() (translate.storage.csvl10n.csvfile</pre>
removeunit() (translate.storage.subtitles.SubRipFile method), 651	require_index() (translate.storage.dtd.dtdfile method), 428
removeunit() (trans-	<pre>require_index() (translate.storage.html.htmlfile</pre>
late.storage.subtitles.SubStationAlphaFile method), 653	method), 437 require_index() (trans-
removeunit () (translate.storage.subtitles.SubtitleFile method), 655	late.storage.html.POHTMLParser method), 435
removeunit() (translate.storage.tbx.tbxfile method), 660	require_index() (translate.storage.ical.icalfile method), 442
removeunit() (translate.storage.tiki.TikiStore method), 666	require_index() (translate.storage.ini.inifile method), 447
removeunit() (translate.storage.tmx.tmxfile method),	require_index() (trans-
671 removeunit() (trans-	late.storage.jsonl10n.ARBJsonFile method), 453
late.storage.trados.TradosTxtTmFile method),	
680	-
removeunit() (translate.storage.ts2.tsfile method),	late.storage.jsonl10n.GoI18NJsonFile method), 458
682	require_index() (trans-
removeunit() (translate.storage.txt.TxtFile method), 688	late.storage.jsonl10n.118NextFile method), 463
<pre>removeunit() (translate.storage.utx.UtxFile method),</pre>	require_index() (trans-
693	late.storage.jsonl10n.JsonFile method), 468
removeunit() (trans-	require_index() (trans-
late.storage.wordfast.WordfastTMFile method), 699	late.storage.jsonl10n.JsonNestedFile method), 470
1.00 1.001 1 1	
removeunit()(translate.storage.xliff.xlifffile method),	require_index() (trans-
removeunit() (translate.storage.xliff.xlifffile method), 706	require_index() (trans- late.storage.jsonl10n.WebExtensionJsonFile
706 renderer (translate.storage.placeables.strelem.StringEle	late.storage.jsonl10n.WebExtensionJsonFile method), 478
706 renderer (translate.storage.placeables.strelem.StringEleattribute), 539	late.storage.jsonl10n.WebExtensionJsonFile em method), 478 require_index() (translate.storage.lisa.LISAfile
706 renderer (translate.storage.placeables.strelem.StringEleattribute), 539 replace() (translate.misc.multistring.multistring	late.storage.jsonl10n.WebExtensionJsonFile em method), 478 require_index() (translate.storage.lisa.LISAfile method), 483
706 renderer (translate.storage.placeables.strelem.StringEleattribute), 539 replace() (translate.misc.multistring.multistring method), 392	late.storage.jsonl10n.WebExtensionJsonFile em method), 478 require_index() (translate.storage.lisa.LISAfile method), 483 require_index() (translate.storage.mo.mofile
706 renderer (translate.storage.placeables.strelem.StringEleattribute), 539 replace() (translate.misc.multistring.multistring method), 392 replace_dom_text() (in module trans-	late.storage.jsonl10n.WebExtensionJsonFile em method), 478 require_index() (translate.storage.lisa.LISAfile method), 483 require_index() (translate.storage.mo.mofile method), 489
706 renderer (translate.storage.placeables.strelem.StringEleattribute), 539 replace() (translate.misc.multistring.multistring method), 392 replace_dom_text() (in module translate.storage.xml_extract.generate), 712	late.storage.jsonl10n.WebExtensionJsonFile em method), 478 require_index() (translate.storage.lisa.LISAfile method), 483 require_index() (translate.storage.mo.mofile method), 489 require_index() (translate.storage.mo.mofile
706 renderer (translate.storage.placeables.strelem.StringEleattribute), 539 replace() (translate.misc.multistring.multistring method), 392 replace_dom_text() (in module trans-	late.storage.jsonl10n.WebExtensionJsonFile em method), 478 require_index() (translate.storage.lisa.LISAfile method), 483 require_index() (translate.storage.mo.mofile method), 489
706 renderer (translate.storage.placeables.strelem.StringEleattribute), 539 replace() (translate.misc.multistring.multistring method), 392 replace_dom_text() (in module translate.storage.xml_extract.generate), 712 ReplacementPlaceable (class in translate.storage.xml_extract.generate)	late.storage.jsonl10n.WebExtensionJsonFile em method), 478 require_index() (translate.storage.lisa.LISAfile method), 483 require_index() (translate.storage.mo.mofile method), 489 require_index() (translate.storage.mo.mofile method), 489
706 renderer (translate.storage.placeables.strelem.StringEleattribute), 539 replace() (translate.misc.multistring.multistring method), 392 replace_dom_text() (in module translate.storage.xml_extract.generate), 712 ReplacementPlaceable (class in translate.storage.placeables.interfaces), 533	late.storage.jsonl10n.WebExtensionJsonFile em method), 478 require_index() (translate.storage.lisa.LISAfile method), 483 require_index() (translate.storage.mo.mofile method), 489 require_index() (translate.storage.mo.mofile late.storage.mozilla_lang.LangStore method), 494
706 renderer (translate.storage.placeables.strelem.StringEleattribute), 539 replace() (translate.misc.multistring.multistring method), 392 replace_dom_text() (in module translate.storage.xml_extract.generate), 712 ReplacementPlaceable (class in translate.storage.placeables.interfaces), 533 Replacer (class in translate.convert.convert), 244 replacestrings() (in module trans-	late.storage.jsonl10n.WebExtensionJsonFile em method), 478 require_index() (translate.storage.lisa.LISAfile method), 483 require_index() (translate.storage.mo.mofile method), 489 require_index() (translate.storage.mo.mofile late.storage.mozilla_lang.LangStore method), 494 require_index() (translate.storage.mozilla_lang.LangStore method), late.storage.omegat.OmegaTFile method),
706 renderer (translate.storage.placeables.strelem.StringEleattribute), 539 replace () (translate.misc.multistring.multistring method), 392 replace_dom_text() (in module translate.storage.xml_extract.generate), 712 ReplacementPlaceable (class in translate.storage.placeables.interfaces), 533 Replacer (class in translate.convert.convert), 244 replacestrings() (in module translate.convert.csv2po), 245	late.storage.jsonl10n.WebExtensionJsonFile em method), 478 require_index() (translate.storage.lisa.LISAfile method), 483 require_index() (translate.storage.mo.mofile method), 489 require_index() (translate.storage.mo.mofile late.storage.mozilla_lang.LangStore method), 494 require_index() (translate.storage.mo.mofile method), 494 require_index() (translate.storage.mozilla_lang.LangStore method), 500
renderer (translate.storage.placeables.strelem.StringEleattribute), 539 replace() (translate.misc.multistring.multistring method), 392 replace_dom_text() (in module translate.storage.xml_extract.generate), 712 ReplacementPlaceable (class in translate.storage.placeables.interfaces), 533 Replacer (class in translate.convert.convert), 244 replacestrings() (in module translate.convert.csv2po), 245 represents_missing() (translate.storage.properties.propunit class method),	late.storage.jsonl10n.WebExtensionJsonFile em method), 478 require_index() (translate.storage.lisa.LISAfile method), 483 require_index() (translate.storage.mo.mofile method), 489 require_index() (translate.storage.mo.mofile ate.storage.mozilla_lang.LangStore method), 494 require_index() (translate.storage.mozilla_lang.LangStore method), 500 require_index() (translate.storage.omegat.OmegaTFile method), 500 require_index() (translate.storage.omegat.OmegaTFileTab method),
renderer (translate.storage.placeables.strelem.StringEleattribute), 539 replace() (translate.misc.multistring.multistring method), 392 replace_dom_text() (in module translate.storage.xml_extract.generate), 712 ReplacementPlaceable (class in translate.storage.placeables.interfaces), 533 Replacer (class in translate.convert.convert), 244 replacestrings() (in module translate.convert.csv2po), 245 represents_missing() (translate.storage.properties.propunit class method), 612	late.storage.jsonl10n.WebExtensionJsonFile em method), 478 require_index() (translate.storage.lisa.LISAfile method), 483 require_index() (translate.storage.mo.mofile method), 489 require_index() (translate.storage.mo.mofile method), 494 require_index() (translate.storage.mozilla_lang.LangStore method), 494 require_index() (translate.storage.omegat.OmegaTFile method), 500 require_index() (translate.storage.omegat.OmegaTFileTab method), 502
renderer (translate.storage.placeables.strelem.StringEleattribute), 539 replace () (translate.misc.multistring.multistring method), 392 replace_dom_text() (in module translate.storage.xml_extract.generate), 712 ReplacementPlaceable (class in translate.storage.placeables.interfaces), 533 Replacer (class in translate.convert.convert), 244 replacestrings() (in module translate.convert.csv2po), 245 represents_missing() (translate.storage.properties.propunit class method), 612 represents_missing() (translate.storage.properties.xwikiunit class method),	late.storage.jsonl10n.WebExtensionJsonFile em method), 478 require_index() (translate.storage.lisa.LISAfile
renderer (translate.storage.placeables.strelem.StringEleattribute), 539 replace () (translate.misc.multistring.multistring method), 392 replace_dom_text() (in module translate.storage.xml_extract.generate), 712 ReplacementPlaceable (class in translate.storage.placeables.interfaces), 533 Replacer (class in translate.convert.convert), 244 replacestrings() (in module translate.convert.csv2po), 245 represents_missing() (translate.storage.properties.propunit class method), 612 represents_missing() (translate.storage.properties.xwikiunit class method), 621 require_index() (translate.storage.base.DictStore	late.storage.jsonl10n.WebExtensionJsonFile em method), 478 require_index() (translate.storage.lisa.LISAfile method), 483 require_index() (translate.storage.mo.mofile method), 489 require_index() (trans- late.storage.mozilla_lang.LangStore method), 494 require_index() (trans- late.storage.omegat.OmegaTFile method), 500 require_index() (trans- late.storage.omegat.OmegaTFileTab method), 502 require_index() (trans- late.storage.php.LaravelPHPFile method), 559
renderer (translate.storage.placeables.strelem.StringEleattribute), 539 replace () (translate.misc.multistring.multistring method), 392 replace_dom_text() (in module translate.storage.xml_extract.generate), 712 ReplacementPlaceable (class in translate.storage.placeables.interfaces), 533 Replacer (class in translate.convert.convert), 244 replacestrings() (in module translate.convert.csv2po), 245 represents_missing() (translate.storage.properties.propunit class method), 612 represents_missing() (translate.storage.properties.xwikiunit class method), 621 require_index() (translate.storage.base.DictStore method), 405	late.storage.jsonl10n.WebExtensionJsonFile em method), 478 require_index() (translate.storage.lisa.LISAfile method), 483 require_index() (translate.storage.mo.mofile method), 489 require_index() (translate.storage.mo.mofile method), 494 require_index() (translate.storage.mozilla_lang.LangStore method), 494 require_index() (translate.storage.omegat.OmegaTFile method), 500 require_index() (translate.storage.omegat.OmegaTFileTab method), 502 require_index() (translate.storage.php.phpfile method), 559 require_index() (translate.storage.php.phpfile method), 564
renderer (translate.storage.placeables.strelem.StringEleattribute), 539 replace () (translate.misc.multistring.multistring method), 392 replace_dom_text() (in module translate.storage.xml_extract.generate), 712 ReplacementPlaceable (class in translate.storage.placeables.interfaces), 533 Replacer (class in translate.convert.convert), 244 replacestrings() (in module translate.convert.csv2po), 245 represents_missing() (translate.storage.properties.propunit class method), 612 represents_missing() (translate.storage.base.DictStore method), 405 require_index() (translate.storage.base.DictStore method), 405 require_index() (translate.storage.base.DictStore method), 405	late.storage.jsonl10n.WebExtensionJsonFile em method), 478 require_index() (translate.storage.lisa.LISAfile method), 483 require_index() (translate.storage.mo.mofile method), 489 require_index() (translate.storage.mo.mofile method), 494 require_index() (translate.storage.mozilla_lang.LangStore method), 494 require_index() (translate.storage.omegat.OmegaTFile method), 500 require_index() (translate.storage.omegat.OmegaTFileTab method), 502 require_index() (translate.storage.php.phpfile method), 559 require_index() (translate.storage.php.phpfile method), 564 require_index() (translate.storage.php.phpfile
renderer (translate.storage.placeables.strelem.StringEleattribute), 539 replace () (translate.misc.multistring.multistring method), 392 replace_dom_text() (in module translate.storage.xml_extract.generate), 712 ReplacementPlaceable (class in translate.storage.placeables.interfaces), 533 Replacer (class in translate.convert.convert), 244 replacestrings() (in module translate.convert.csv2po), 245 represents_missing() (translate.storage.properties.propunit class method), 612 represents_missing() (translate.storage.base.DictStore method), 405 require_index() (translate.storage.base.DictStore method), 405 require_index() (translate.storage.base.TranslationStore method),	late.storage.jsonl10n.WebExtensionJsonFile em method), 478 require_index() (translate.storage.lisa.LISAfile method), 483 require_index() (translate.storage.mo.mofile method), 489 require_index() (translate.storage.mo.mofile method), 494 require_index() (translate.storage.mozilla_lang.LangStore method), 494 require_index() (translate.storage.omegat.OmegaTFile method), 500 require_index() (translate.storage.omegat.OmegaTFileTab method), 502 require_index() (translate.storage.php.phpfile method), 564 require_index() (translate.storage.php.phpfile method), 564 require_index() (translate.storage.php.phpfile method), 564 require_index() (translate.storage.php.phpfile method), 570
renderer (translate.storage.placeables.strelem.StringEleattribute), 539 replace () (translate.misc.multistring.multistring method), 392 replace_dom_text() (in module translate.storage.xml_extract.generate), 712 ReplacementPlaceable (class in translate.storage.placeables.interfaces), 533 Replacer (class in translate.convert.convert), 244 replacestrings() (in module translate.convert.csv2po), 245 represents_missing() (translate.storage.properties.propunit class method), 612 represents_missing() (translate.storage.base.DictStore method), 405 require_index() (translate.storage.base.DictStore method), 405 require_index() (translate.storage.base.DictStore method), 410	late.storage.jsonl10n.WebExtensionJsonFile em method), 478 require_index() (translate.storage.lisa.LISAfile method), 483 require_index() (translate.storage.mo.mofile method), 489 require_index() (translate.storage.mo.mofile method), 494 require_index() (translate.storage.mozilla_lang.LangStore method), 494 require_index() (translate.storage.omegat.OmegaTFile method), 500 require_index() (translate.storage.omegat.OmegaTFileTab method), 502 require_index() (translate.storage.php.phpfile method), 559 require_index() (translate.storage.php.phpfile method), 564 require_index() (translate.storage.php.phpfile method), 564 require_index() (translate.storage.php.phpfile method), 570 require_index() (translate.storage.php.phpfile method), 570 require_index() (translate.storage.php.phpfile method), 570

require_index() (trans-	method), 660
late.storage.properties.javafile method), 599	require_index() (translate.storage.tiki.TikiStore method), 666
require_index() (trans-	require_index() (translate.storage.tmx.tmxfile
late.storage.properties.javautf16file method), 601	<pre>method), 671 require_index() (trans-</pre>
require_index() (trans-	late.storage.trados.TradosTxtTmFile method),
late.storage.properties.javautf8file method),	680
603 require_index() (trans-	require_index() (translate.storage.ts2.tsfile method), 682
late.storage.properties.joomlafile method),	require_index() (translate.storage.txt.TxtFile
604	method), 688
require_index() (trans- late.storage.properties.propfile method),	require_index() (translate.storage.utx.UtxFile method), 693
606	require_index() (trans-
require_index() (trans-	late.storage.wordfast.WordfastTMFile method),
late.storage.properties.stringsfile method), 614	699 require_index() (translate.storage.xliff.xlifffile
require_index() (trans-	method), 706
late.storage.properties.stringsutf8file method), 616	reset() (translate.misc.ourdom.ExpatBuilderNS method), 397
require_index() (trans-	reset () (translate.storage.html.htmlfile method), 437
late.storage.properties.xwikifile method),	reset() (translate.storage.html.POHTMLParser
618 require_index() (trans-	method), 435 resurrect() (translate.storage.pypo.pounit method),
late.storage.properties.XWikiFullPage	627
method), 593	resx2po (class in translate.convert.resx2po), 269
require_index() (trans- late.storage.properties.XWikiPageProperties	rfind() (translate.misc.multistring.multistring method), 392
method), 595	rich_parsers (trans-
require_index() (translate.storage.pypo.pofile method), 624	$late. storage. base. Translation Unit \qquad attribute),$
	413
require_index() (translate.storage.qm.qmfile	rich_source (translate.storage.base.DictUnit at-
require_index() (translate.storage.qm.qmfile method), 630	rich_source (translate.storage.base.DictUnit attribute), 408
require_index() (translate.storage.qm.qmfile	rich_source (translate.storage.base.DictUnit at-
require_index() method), 630 require_index() method), 635 require_index() require_index() (translate.storage.qm.qmfile (translate.storage.qph.QphFile (translate.storage.rc.rcfile	rich_source (translate.storage.base.DictUnit at- tribute), 408 rich_source (translate.storage.base.TranslationUnit attribute), 413 rich_source (translate.storage.catkeys.CatkeysUnit
require_index() method), 630 require_index() method), 635 require_index() method), 641 (translate.storage.qm.qmfile (translate.storage.qph.QphFile (translate.storage.rc.rcfile	rich_source (translate.storage.base.DictUnit at- tribute), 408 rich_source (translate.storage.base.TranslationUnit attribute), 413 rich_source (translate.storage.catkeys.CatkeysUnit attribute), 420
require_index() method), 630 require_index() method), 635 require_index() method), 641 require_index() method), 641 require_index() late.storage.qm.qmfile (translate.storage.qph.QphFile (translate.storage.rc.rcfile (translate.storage.rc.rcfile (translate.storage.rc.rcfile	rich_source (translate.storage.base.DictUnit at- tribute), 408 rich_source (translate.storage.base.TranslationUnit attribute), 413 rich_source (translate.storage.catkeys.CatkeysUnit attribute), 420 rich_source (translate.storage.csvl10n.csvunit attribute), 425
require_index() (translate.storage.qm.qmfile method), 630 require_index() (translate.storage.qph.QphFile method), 635 require_index() (translate.storage.rc.rcfile method), 641 require_index() (translate.storage.rc.rcfile method), 648	rich_source (translate.storage.base.DictUnit at- tribute), 408 rich_source (translate.storage.base.TranslationUnit attribute), 413 rich_source (translate.storage.catkeys.CatkeysUnit attribute), 420 rich_source (translate.storage.csvl10n.csvunit attribute), 425 rich_source (translate.storage.dtd.dtdunit attribute),
require_index() (translate.storage.qm.qmfile method), 630 require_index() (translate.storage.qph.QphFile method), 635 require_index() (translate.storage.rc.rcfile method), 641 require_index() (translate.storage.rc.rcfile method), 648 require_index() (translate.storage.rc.rcfile (translate.storage.rc.rcfile (translate.storage.rc.rcfile (translate.storage.rc.rcfile (translate.storage.rc.rcfile (translate.storage.rc.rcfile (translate.storage.qph.QphFile (translate	rich_source (translate.storage.base.DictUnit at- tribute), 408 rich_source (translate.storage.base.TranslationUnit attribute), 413 rich_source (translate.storage.catkeys.CatkeysUnit attribute), 420 rich_source (translate.storage.csvl10n.csvunit attribute), 425 rich_source (translate.storage.dtd.dtdunit attribute), 431
require_index() (translate.storage.qm.qmfile method), 630 require_index() (translate.storage.qph.QphFile method), 635 require_index() (translate.storage.rc.rcfile method), 641 require_index() (translate.storage.rc.rcfile method), 648 require_index() (translate.storage.subtitles.AdvSubStationAlphaFile method), 648 require_index() (translate.storage.subtitles.AdvSubStationAlphaFile method), 650	rich_source (translate.storage.base.DictUnit attribute), 408 rich_source (translate.storage.base.TranslationUnit attribute), 413 rich_source (translate.storage.catkeys.CatkeysUnit attribute), 420 rich_source (translate.storage.csvl10n.csvunit attribute), 425 rich_source (translate.storage.dtd.dtdunit attribute), 431 rich_source (translate.storage.html.htmlunit attribute), 440
require_index() (translate.storage.qm.qmfile method), 630 require_index() (translate.storage.qph.QphFile method), 635 require_index() (translate.storage.rc.rcfile method), 641 require_index() (translate.storage.rc.rcfile method), 648 require_index() (translate.storage.subtitles.AdvSubStationAlphaFile method), 648 require_index() (translate.storage.subtitles.AdvSubStationAlphaFile method), 650 require_index() (translate.storage.qph.QphFile method), 648	rich_source (translate.storage.base.DictUnit attribute), 408 rich_source (translate.storage.base.TranslationUnit attribute), 413 rich_source (translate.storage.catkeys.CatkeysUnit attribute), 420 rich_source (translate.storage.csvl10n.csvunit attribute), 425 rich_source (translate.storage.dtd.dtdunit attribute), 431 rich_source (translate.storage.html.htmlunit attribute), 440 rich_source (translate.storage.ical.icalunit at-
require_index() (translate.storage.qm.qmfile method), 630 require_index() (translate.storage.qph.QphFile method), 635 require_index() (translate.storage.rc.rcfile method), 641 require_index() (translate.storage.rc.rcfile method), 648 require_index() (translate.storage.subtitles.AdvSubStationAlphaFile method), 648 require_index() (translate.storage.subtitles.AdvSubStationAlphaFile method), 650	rich_source (translate.storage.base.DictUnit attribute), 408 rich_source (translate.storage.base.TranslationUnit attribute), 413 rich_source (translate.storage.catkeys.CatkeysUnit attribute), 420 rich_source (translate.storage.csvl10n.csvunit attribute), 425 rich_source (translate.storage.dtd.dtdunit attribute), 431 rich_source (translate.storage.html.htmlunit attribute), 440 rich_source (translate.storage.ical.icalunit attribute), 445
require_index() (translate.storage.qm.qmfile method), 630 require_index() (translate.storage.qph.QphFile method), 635 require_index() (translate.storage.rc.rcfile method), 641 require_index() (translate.storage.rc.rcfile method), 648 require_index() (translate.storage.subtitles.AdvSubStationAlphaFile method), 648 require_index() (translate.storage.subtitles.AdvSubStationAlphaFile method), 650 require_index() (translate.storage.subtitles.MicroDVDFile method), 650 require_index() (translate.storage.subtitles.SubRipFile method), 651 require_index() (translate.storage.rc.rcfile (translate.storage.subtitles.MicroDVDFile method), 651 require_index() (translate.storage.rc.rcfile (translate.storage.subtitles.MicroDVDFile method), 651	rich_source (translate.storage.base.DictUnit attribute), 408 rich_source (translate.storage.base.TranslationUnit attribute), 413 rich_source (translate.storage.catkeys.CatkeysUnit attribute), 420 rich_source (translate.storage.csvl10n.csvunit attribute), 425 rich_source (translate.storage.dtd.dtdunit attribute), 431 rich_source (translate.storage.html.htmlunit attribute), 440 rich_source (translate.storage.ical.icalunit attribute), 445 rich_source (translate.storage.ini.iniunit attribute), 445
require_index() (translate.storage.qm.qmfile method), 630 require_index() (translate.storage.qph.QphFile method), 635 require_index() (translate.storage.rc.rcfile method), 641 require_index() (translate.storage.rc.rcfile method), 648 require_index() (translate.storage.subtitles.AdvSubStationAlphaFile method), 648 require_index() (translate.storage.subtitles.MicroDVDFile method), 650 require_index() (translate.storage.subtitles.MicroDVDFile method), 651 require_index() (translate.storage.subtitles.SubRipFile method), 651 require_index() (translate.storage.subtitles.SubStationAlphaFile	rich_source (translate.storage.base.DictUnit attribute), 408 rich_source (translate.storage.base.TranslationUnit attribute), 413 rich_source (translate.storage.catkeys.CatkeysUnit attribute), 420 rich_source (translate.storage.csvl10n.csvunit attribute), 425 rich_source (translate.storage.dtd.dtdunit attribute), 431 rich_source (translate.storage.html.htmlunit attribute), 440 rich_source (translate.storage.ical.icalunit attribute), 445 rich_source (translate.storage.ini.iniunit attribute), 450 rich_source (translate.storage.ini.iniunit attribute), 150 rich_source (translate.storage.ini.iniunit attribute), 150 rich_source (translate.storage.ini.iniunit attribute), 150 rich_source (translate.storage.ini.iniunit attribute), 150
require_index() (translate.storage.qm.qmfile method), 630 require_index() (translate.storage.qph.QphFile method), 635 require_index() (translate.storage.rc.rcfile method), 641 require_index() (translate.storage.rc.rcfile method), 648 require_index() (translate.storage.subtitles.AdvSubStationAlphaFile method), 648 require_index() (translate.storage.subtitles.AdvSubStationAlphaFile method), 650 require_index() (translate.storage.subtitles.MicroDVDFile method), 650 require_index() (translate.storage.subtitles.SubRipFile method), 651 require_index() (translate.storage.rc.rcfile (translate.storage.subtitles.MicroDVDFile method), 651 require_index() (translate.storage.rc.rcfile (translate.storage.subtitles.MicroDVDFile method), 651	rich_source (translate.storage.base.DictUnit attribute), 408 rich_source (translate.storage.base.TranslationUnit attribute), 413 rich_source (translate.storage.catkeys.CatkeysUnit attribute), 420 rich_source (translate.storage.csvl10n.csvunit attribute), 425 rich_source (translate.storage.dtd.dtdunit attribute), 431 rich_source (translate.storage.html.htmlunit attribute), 440 rich_source (translate.storage.ical.icalunit attribute), 445 rich_source (translate.storage.ini.iniunit attribute), 445
require_index() (translate.storage.qm.qmfile method), 630 require_index() (translate.storage.qph.QphFile method), 635 require_index() (translate.storage.rc.rcfile method), 641 require_index() (translate.storage.rc.rcfile method), 648 require_index() (translate.storage.subtitles.AdvSubStationAlphaFile method), 648 require_index() (translate.storage.subtitles.MicroDVDFile method), 650 require_index() (translate.storage.subtitles.MicroDVDFile method), 651 require_index() (translate.storage.subtitles.SubRipFile method), 651 require_index() (translate.storage.subtitles.SubStationAlphaFile method), 653 require_index() (translate.storage.subtitles.SubStationAlphaFile method), 653 require_index() (translate.storage.subtitles.SubtitleFile method), 653	rich_source (translate.storage.base.DictUnit attribute), 408 rich_source (translate.storage.base.TranslationUnit attribute), 413 rich_source (translate.storage.catkeys.CatkeysUnit attribute), 420 rich_source (translate.storage.csvl10n.csvunit attribute), 425 rich_source (translate.storage.dtd.dtdunit attribute), 431 rich_source (translate.storage.html.htmlunit attribute), 440 rich_source (translate.storage.ical.icalunit attribute), 445 rich_source (translate.storage.ini.iniunit attribute), 450 rich_source (translate.storage.ini.iniunit attribute), 456 rich_source (translate.storage.ini.iniunit attribute), 456 rich_source (translate.storage.ini.ounit attribute), 456 rich_source (translate.storage.ini.ounit attribute), 456
require_index() (translate.storage.qm.qmfile method), 630 require_index() (translate.storage.qph.QphFile method), 635 require_index() (translate.storage.rc.rcfile method), 641 require_index() (translate.storage.rc.rcfile method), 648 require_index() (translate.storage.subtitles.AdvSubStationAlphaFile method), 648 require_index() (translate.storage.subtitles.AdvSubStationAlphaFile method), 650 require_index() (translate.storage.subtitles.SubStationAlphaFile method), 651 require_index() (translate.storage.subtitles.SubStationAlphaFile method), 653 require_index() (translate.storage.subtitles.SubStationAlphaFile method), 653 require_index() (translate.storage.qph.QphFile method), 653 require_index() (translate.storage.rc.rcfile (translate.storage.subtitles.SubStationAlphaFile method), 653	rich_source (translate.storage.base.DictUnit attribute), 408 rich_source (translate.storage.base.TranslationUnit attribute), 413 rich_source (translate.storage.catkeys.CatkeysUnit attribute), 420 rich_source (translate.storage.csvl10n.csvunit attribute), 425 rich_source (translate.storage.dtd.dtdunit attribute), 431 rich_source (translate.storage.html.htmlunit attribute), 440 rich_source (translate.storage.ical.icalunit attribute), 445 rich_source (translate.storage.ini.iniunit attribute), 450 rich_source (translate.storage.ini.iniunit attribute), 450 rich_source (translate.storage.ini.iniunit attribute), 456

- attribute), 466
- rich source (translate.storage.jsonl10n.JsonNestedUnit attribute), 473
- rich source (translate.storage.jsonl10n.JsonUnit attribute), 476
- rich source (translate.storage.jsonl10n.WebExtensionJsonUnit attribute), 481
- rich_source (translate.storage.lisa.LISAunit *tribute*), 486
- rich_source (translate.storage.mo.mounit attribute), 492
- rich_source (translate.storage.mozilla_lang.LangUnit attribute), 497
- rich_source (translate.storage.omegat.OmegaTUnit attribute), 505
- rich source (translate.storage.php.LaravelPHPUnit attribute), 562
- (translate.storage.php.phpunit rich source attribute), 567
- rich source (translate.storage.pocommon.pounit at*tribute*), 573
- rich_source (translate.storage.poxliff.PoXliffUnit attribute), 582
- rich_source (translate.storage.properties.proppluralunit attribute), 609
- rich_source (translate.storage.properties.propunit attribute), 612
- rich_source (translate.storage.properties.xwikiunit attribute), 621
- (translate.storage.pypo.pounit rich source attribute), 627
- rich_source (translate.storage.qm.qmunit attribute), 633
- rich source (translate.storage.qph.QphUnit tribute), 638
- rich source (translate.storage.rc.rcunit attribute), 644
- rich source (translate.storage.subtitles.SubtitleUnit attribute), 658
- rich_source (translate.storage.tbx.tbxunit attribute), 663
- rich_source (translate.storage.tiki.TikiUnit at*tribute*), 669
- rich_source (translate.storage.tmx.tmxunit attribute), 674
- rich_source (translate.storage.trados.TradosUnit at*tribute*), 678
- rich_source (translate.storage.ts2.tsunit attribute), rich_target 685

- rich_source (translate.storage.jsonl10n.118NextUnit rich_source (translate.storage.txt.TxtUnit attribute), 691
 - rich source (translate.storage.utx.UtxUnit *tribute*), 696
 - rich source (translate.storage.wordfast.WordfastUnit attribute), 703
 - rich source (translate.storage.xliff.xliffunit tribute), 710
 - rich_target (translate.storage.base.DictUnit attribute), 408
 - rich_target (translate.storage.base.TranslationUnit attribute), 413
 - rich_target (translate.storage.catkeys.CatkeysUnit attribute), 420
 - rich_target (translate.storage.csvl10n.csvunit attribute), 425
 - rich_target (translate.storage.dtd.dtdunit attribute), 431
 - rich target (translate.storage.html.htmlunit tribute), 440
 - (translate.storage.ical.icalunit rich target *tribute*), 446
 - rich target (translate.storage.ini.iniunit attribute), 451
 - rich_target (translate.storage.jsonl10n.ARBJsonUnit attribute), 456
 - rich_target (translate.storage.jsonl10n.GoI18NJsonUnit at*tribute*), 461
 - rich_target (translate.storage.jsonl10n.I18NextUnit attribute), 466
 - (transrich_target late.storage.jsonl10n.JsonNestedUnit at*tribute*), 473
 - rich target (translate.storage.jsonl10n.JsonUnit attribute), 476
 - rich_target (translate.storage.jsonl10n.WebExtensionJsonUnit attribute), 481
 - rich target (translate.storage.lisa.LISAunit tribute), 487
 - rich_target (translate.storage.mo.mounit attribute), 493
 - rich_target (translate.storage.mozilla_lang.LangUnit attribute),
 - rich_target (translate.storage.omegat.OmegaTUnit attribute), 505
 - rich_target (translate.storage.php.LaravelPHPUnit attribute), 562
 - (translate.storage.php.phpunit tribute), 567

rich_target (translate.storage.pocommon.pounit at-	
tribute), 573	late.storage.html.htmlunit class method),
rich_target (translate.storage.poxliff.PoXliffUnit at-	440
tribute), 582 rich_target (trans-	rich_to_multistring() (trans- late.storage.ical.icalunit class method),
late.storage.properties.proppluralunit at-	446
tribute), 609	rich_to_multistring() (trans-
$\verb rich_target & \textit{(translate.storage.properties.propunit)} \\$	late.storage.ini.iniunit class method), 451
attribute), 612	rich_to_multistring() (trans-
rich_target (translate.storage.properties.xwikiunit attribute), 621	late.storage.jsonl10n.ARBJsonUnit class method), 456
rich_target (translate.storage.pypo.pounit at- tribute), 627	rich_to_multistring() (trans- late.storage.jsonl10n.GoI18NJsonUnit class
$\verb rich_target (translate.storage.qm.qmunit attribute),$	method), 461
633	rich_to_multistring() (trans-
rich_target (translate.storage.qph.QphUnit attribute), 639	late.storage.jsonl10n.I18NextUnit class method), 466
rich_target (translate.storage.rc.rcunit attribute), 644	rich_to_multistring() (trans- late.storage.jsonl10n.JsonNestedUnit class
rich_target (translate.storage.subtitles.SubtitleUnit	method), 473
attribute), 658	rich_to_multistring() (trans-
rich_target (translate.storage.tbx.tbxunit attribute), 663	late.storage.jsonl10n.JsonUnit class method), 476
rich_target (translate.storage.tiki.TikiUnit at-	rich_to_multistring() (trans-
tribute), 669	late.storage.jsonl10n.WebExtensionJsonUnit
rich_target (translate.storage.tmx.tmxunit at- tribute), 674	<pre>class method), 481 rich_to_multistring() (trans-</pre>
rich_target (translate.storage.trados.TradosUnit at- tribute), 678	late.storage.lisa.LISAunit class method), 487
rich_target (translate.storage.ts2.tsunit attribute), 686	rich_to_multistring() (trans- late.storage.mo.mounit class method), 493
<pre>rich_target (translate.storage.txt.TxtUnit attribute),</pre>	rich_to_multistring() (trans-
691 rich_target (translate.storage.utx.UtxUnit at-	late.storage.mozilla_lang.LangUnit class method), 498
tribute), 696	rich_to_multistring() (trans-
rich_target (trans-	late.storage.omegat.OmegaTUnit class
late.storage.wordfast.WordfastUnit attribute),	method), 505
703	rich_to_multistring() (trans-
rich_target (translate.storage.xliff.xliffunit at- tribute), 710	late.storage.php.LaravelPHPUnit class method), 562
rich_to_multistring() (trans-	rich_to_multistring() (trans-
late.storage.base.DictUnit class method), 408	late.storage.php.phpunit class method), 567
rich_to_multistring() (trans-	rich_to_multistring() (trans-
late.storage.base.TranslationUnit class method), 413	late.storage.pocommon.pounit class method), 573
rich_to_multistring() (trans-	rich_to_multistring() (trans-
late.storage.catkeys.CatkeysUnit class method), 420	late.storage.poxliff.PoXliffUnit class method), 583
rich_to_multistring() (trans-	rich_to_multistring() (trans-
late.storage.csvl10n.csvunit class method), 425	late.storage.properties.proppluralunit class method), 609
rich_to_multistring() (trans-	`
late.storage.dtd.dtdunit class method), 431	late.storage.properties.propunit class method),

612	method), 240
rich_to_multistring() (trans-	run () (translate.convert.convert.ConvertOptionParser
late.storage.properties.xwikiunit class method),	method), 243
621	run () (translate.convert.ical2po.ical2po method), 247
rich_to_multistring() (trans-	run () (translate.convert.ini2po.ini2po method), 247
late.storage.pypo.pounit class method), 627	run() (translate.convert.mozlang2po.lang2po method), 249
rich_to_multistring() (trans-	run() (translate.convert.php2po.php2po method), 250
late.storage.qm.qmunit class method), 633	run() (translate.convert.po2ical.po2ical method), 251
rich_to_multistring() (trans-	run () (translate.convert.po2ini.po2ini method), 252
late.storage.qph.QphUnit class method), 639	run () (translate.convert.po2moz.MozConvertOptionParser method), 255
rich_to_multistring() (trans-	run() (translate.convert.po2mozlang.po2lang method),
late.storage.rc.rcunit class method), 644	252
rich_to_multistring() (trans-	run () (translate.convert.po2tiki.po2tiki method), 258
late.storage.subtitles.SubtitleUnit class method), 658	run() (translate.convert.po2tmx.TmxOptionParser method), 261
rich_to_multistring() (trans-	run() (translate.convert.po2txt.po2txt method), 262
late.storage.tbx.tbxunit class method), 663	run () (translate.convert.po2wordfast.WfOptionParser
rich_to_multistring() (trans-	method), 265
late.storage.tiki.TikiUnit class method), 669	run() (translate.convert.po2yaml.po2yaml method), 267
rich_to_multistring() (trans-	run () (translate.convert.tiki2po.tiki2po method), 270
late.storage.tmx.tmxunit class method), 674	run() (translate.convert.txt2po.txt2po method), 271
rich_to_multistring() (trans-	run() (translate.convert.yaml2po.yaml2po method),
late.storage.trados.TradosUnit class method),	272
678	run () (translate.filters.pofilter.FilterOptionParser
rich_to_multistring() (trans-	method), 355
late.storage.ts2.tsunit class method), 686	run () (translate.misc.optrecurse.RecursiveOptionParser
rich_to_multistring() (trans-	method), 396
late.storage.txt.TxtUnit class method), 691	run() (translate.tools.poconflicts.ConflictOptionParser
rich_to_multistring() (trans-	method), 717
late.storage.utx.UtxUnit class method), 696	run () (translate.tools.pogrep.GrepOptionParser
rich_to_multistring() (trans-	method), 721
late.storage.wordfast.WordfastUnit class method), 703	run() (translate.tools.porestructure.SplitOptionParser method), 725
rich_to_multistring() (trans- late.storage.xliff.xliffunit class method),	run () (translate.tools.poterminology.TerminologyOptionParser method), 728
710	run_converter() (in module trans-
rindex() (translate.misc.multistring.multistring	late.convert.ical2po), 247
method), 392	run_converter() (in module trans-
rjust() (translate.misc.multistring.multistring	late.convert.ini2po), 247
method), 392	run_converter() (in module trans-
rpartition() (translate.misc.multistring.multistring	late.convert.mozlang2po), 249
method), 392	run_converter() (in module trans-
rsplit() (translate.misc.multistring.multistring	late.convert.php2po), 250
method), 392	run_converter() (in module trans-
rstrip() (translate.misc.multistring.multistring	late.convert.po2ical), 251
method), 392	run_converter() (in module trans-
RTF_ESCAPES (in module translate.storage.trados),	late.convert.po2ini), 252
675	run_converter() (in module trans-
rtlpunc (translate.lang.common.Common attribute),	late.convert.po2mozlang), 252
363	run_converter() (in module trans-
run () (translate.convert.convert.ArchiveConvertOptionPo	arser late.convert.po2tiki), 258

<pre>run_converter() (in module trans-</pre>	method), 350
late.convert.po2txt), 262	<pre>run_test() (translate.filters.checks.CCLicenseChecker</pre>
<pre>run_converter() (in module trans-</pre>	method), 276
late.convert.po2yaml), 267	<pre>run_test() (translate.filters.checks.DrupalChecker</pre>
<pre>run_converter() (in module trans-</pre>	method), 282
late.convert.tiki2po), 270	<pre>run_test() (translate.filters.checks.GnomeChecker</pre>
run_converter() (in module trans-	method), 288
late.convert.txt2po), 271	run_test() (translate.filters.checks.IOSChecker
run_converter() (in module trans-	method), 294
late.convert.yaml2po), 272	run_test() (translate.filters.checks.KdeChecker
run_filters() (trans-	method), 300
late.filters.checks.CCLicenseChecker method),	run_test() (translate.filters.checks.L20nChecker
276	method), 305
run_filters() (trans-	run_test() (translate.filters.checks.LibreOfficeChecker
late.filters.checks.DrupalChecker method),	method), 311
282	run_test() (translate.filters.checks.MinimalChecker
run_filters() (trans-	method), 317
late.filters.checks.GnomeChecker method),	run_test() (translate.filters.checks.MozillaChecker
288	method), 323
<pre>run_filters() (translate.filters.checks.IOSChecker</pre>	<pre>run_test() (translate.filters.checks.OpenOfficeChecker</pre>
method), 294	method), 328
<pre>run_filters() (translate.filters.checks.KdeChecker</pre>	<pre>run_test() (translate.filters.checks.ReducedChecker</pre>
method), 300	method), 334
<pre>run_filters() (translate.filters.checks.L20nChecker</pre>	run_test() (translate.filters.checks.StandardChecker
method), 305	method), 340
run_filters() (trans-	<pre>run_test() (translate.filters.checks.StandardUnitChecker</pre>
late.filters.checks.LibreOfficeChecker method),	method), 343
311	run_test() (translate.filters.checks.TermChecker
<pre>run_filters() (trans-</pre>	method), 347
late.filters.checks.MinimalChecker method),	run_test() (translate.filters.checks.TranslationChecker
317	method), 350
run_filters() (trans-	run_test() (translate.filters.checks.UnitChecker
late.filters.checks.MozillaChecker method),	method), 350
323	runclean() (in module translate.tools.poclean), 715
run_filters() (trans-	runfilter() (in module translate.filters.pofilter), 355
late.filters.checks.OpenOfficeChecker method),	rungrep () (in module translate.tools.pogrep), 722
328	runtests() (in module translate.filters.checks), 351
run_filters() (trans-	S
late.filters.checks.ReducedChecker method),	
334	save() (translate.storage.base.DictStore method), 405
run_filters() (trans-	save() (translate.storage.base.TranslationStore
late.filters.checks.StandardChecker method),	method), 410
340	save() (translate.storage.bundleprojstore.BundleProjectStore
run_filters() (trans-	method), 414
late. filters. checks. Standard Unit Checker	save() (translate.storage.catkeys.CatkeysFile method),
method), 343	417
<pre>run_filters() (translate.filters.checks.TeeChecker</pre>	save() (translate.storage.csvl10n.csvfile method), 422
method), 344	save() (translate.storage.dtd.dtdfile method), 428
<pre>run_filters() (translate.filters.checks.TermChecker</pre>	save() (translate.storage.html.htmlfile method), 437
method), 347	save() (translate.storage.html.POHTMLParser
run_filters() (trans-	method), 435
late.filters.checks.TranslationChecker method),	
350	save () (translate.storage.ical.icalfile method), 442
run_filters() (translate.filters.checks.UnitChecker	save() (translate.storage.ini.inifile method), 448

save() (translate.storage.jsonl10n.ARBJsonFile method), 453	save () (translate.storage.qph.QphFile method), 635
save() (translate.storage.jsonl10n.GoI18NJsonFile	save() (translate.storage.rc.rcfile method), 641 save() (translate.storage.subtitles.AdvSubStationAlphaFile
method), 458	method), 648
save() (translate.storage.jsonl10n.118NextFile method), 463	save() (translate.storage.subtitles.MicroDVDFile method), 650
save() (translate.storage.jsonl10n.JsonFile method), 468	save() (translate.storage.subtitles.SubRipFile method), 651
save() (translate.storage.jsonl10n.JsonNestedFile method), 470	save() (translate.storage.subtitles.SubStationAlphaFile method), 653
save() (translate.storage.jsonl10n.WebExtensionJsonFilmethod), 478	
save() (translate.storage.lisa.LISAfile method), 483	save() (translate.storage.tbx.tbxfile method), 660
save() (translate.storage.mo.mofile method), 489	save() (translate.storage.tiki.TikiStore method), 666
save() (translate.storage.mozilla_lang.LangStore	save() (translate.storage.tmx.tmxfile method), 671
method), 495	save() (translate.storage.trados.TradosTxtTmFile
save() (translate.storage.omegat.OmegaTFile method),	method), 680
500	save() (translate.storage.ts2.tsfile method), 682
save() (translate.storage.omegat.OmegaTFileTab	save() (translate.storage.txt.TxtFile method), 688
method), 502	save () (translate.storage.utx.UtxFile method), 693
save() (translate.storage.php.LaravelPHPFile method), 559	save() (translate.storage.wordfast.WordfastTMFile method), 699
save() (translate.storage.php.phpfile method), 564	save() (translate.storage.xliff.xlifffile method), 706
save() (translate.storage.pocommon.pofile method), 570	savefile() (translate.storage.base.DictStore method), 405
save() (translate.storage.poxliff.PoXliffFile method), 579	<pre>savefile() (translate.storage.base.TranslationStore</pre>
save() (translate.storage.project.Project method), 584	<pre>savefile() (translate.storage.catkeys.CatkeysFile</pre>
save() (translate.storage.projstore.ProjectStore	method), 417
method), 584	savefile() (translate.storage.csvl10n.csvfile
save() (translate.storage.properties.gwtfile method), 597	method), 422
save() (translate.storage.properties.javafile method),	savefile() (translate.storage.dtd.dtdfile method), 428 savefile() (translate.storage.html.htmlfile method),
599	437
save() (translate.storage.properties.javautf16file method), 601	savefile() (translate.storage.html.POHTMLParser method), 435
save() (translate.storage.properties.javautf8file method), 603	savefile() (translate.storage.ical.icalfile method), 443
save() (translate.storage.properties.joomlafile	savefile() (translate.storage.ini.inifile method), 448
method), 604	savefile() (translate.storage.jsonl10n.ARBJsonFile
save() (translate.storage.properties.propfile method),	method), 453
606 (tuanulata ataugga nuonautias atuingafila	savefile() (translate.storage.jsonl10n.GoI18NJsonFile
save() (translate.storage.properties.stringsfile method), 614	method), 458 savefile() (translate.storage.jsonl10n.I18NextFile
save() (translate.storage.properties.stringsutf8file	method), 463
method), 616	savefile() (translate.storage.jsonl10n.JsonFile
save() (translate.storage.properties.xwikifile method),	method), 468
618	savefile() (translate.storage.jsonl10n.JsonNestedFile
save() (translate.storage.properties.XWikiFullPage	method), 470
method), 593	savefile() (translate.storage.jsonl10n.WebExtensionJsonFile
save() (translate.storage.properties.XWikiPagePropertie method), 595	es method), 478 savefile() (translate.storage.lisa.LISAfile method),
save() (translate.storage.pypo.pofile method), 624	483
save() (translate.storage.qm.qmfile method), 630	savefile() (translate.storage.mo.mofile method), 489
	· · · · · · · · · · · · · · · · · · ·

savefile() (translate.storage.mozilla_lang.LangStore method), 495	666 savefile() (translate.storage.tmx.tmxfile method),
savefile() (translate.storage.omegat.OmegaTFile	671
method), 500	savefile() (translate.storage.trados.TradosTxtTmFile
savefile() (translate.storage.omegat.OmegaTFileTab method), 502	method), 680 savefile() (translate.storage.ts2.tsfile method), 682
savefile() (translate.storage.php.LaravelPHPFile	<pre>savefile() (translate.storage.txt.TxtFile method),</pre>
method), 559	688
savefile() (translate.storage.php.phpfile method), 564	savefile() (translate.storage.utx.UtxFile method), 693
savefile() (translate.storage.pocommon.pofile method), 570	<pre>savefile() (translate.storage.wordfast.WordfastTMFile method), 699</pre>
savefile() (translate.storage.poxliff.PoXliffFile method), 579	savefile() (translate.storage.xliff.xlifffile method), 706
savefile() (translate.storage.properties.gwtfile	scanfiles() (translate.storage.directory.Directory
method), 597	method), 426
savefile() (translate.storage.properties.javafile method), 599	scanfiles() (translate.storage.zip.ZIPFile method), 714
<pre>savefile() (translate.storage.properties.javautf16file</pre>	scripts (in module translate.lang.data), 365
method), 601	<pre>searchElementsByTagName_helper() (in mod-</pre>
savefile() (translate.storage.properties.javautf8file	ule translate.misc.ourdom), 398
method), 603 savefile() (translate.storage.properties.joomlafile	searchreplaceinput() (trans- late.convert.convert.Replacer method), 244
method), 604	searchreplacetemplate() (trans-
savefile() (translate.storage.properties.propfile	late.convert.convert.Replacer method), 244
method), 606	segmentfile() (in module trans-
<pre>savefile() (translate.storage.properties.stringsfile</pre>	late.tools.posegment), 725
method), 614	sentence_iter() (translate.lang.af.af class
savefile() (translate.storage.properties.stringsutf8file	method), 357
method), 616 savefile() (translate.storage.properties.xwikifile	sentence_iter() (translate.lang.am.am class method), 358
method), 618	sentence_iter() (translate.lang.ar.ar class
savefile()(translate.storage.properties.XWikiFullPage	
method), 593	sentence_iter() (translate.lang.bn.bn class
savefile() (translate.storage.properties.XWikiPagePro	
method), 595	sentence_iter() (translate.lang.code_or.code_or
savefile() (translate.storage.pypo.pofile method), 624	<pre>class method), 360 sentence_iter() (translate.lang.common.Common</pre>
savefile() (translate.storage.qm.qmfile method), 630	class method), 363
savefile() (translate.storage.qph.QphFile method),	sentence_iter() (translate.lang.de.de class
635	method), 366
savefile() (translate.storage.rc.rcfile method), 641	<pre>sentence_iter() (translate.lang.el.el class method),</pre>
savefile() (translate.storage.subtitles.AdvSubStationA	
method), 648	sentence_iter() (translate.lang.es.es class
savefile() (translate.storage.subtitles.MicroDVDFile method), 650	<pre>method), 367 sentence_iter() (translate.lang.fa.fa class</pre>
savefile() (translate.storage.subtitles.SubRipFile	method), 368
method), 651	sentence_iter() (translate.lang.fi.fi class method),
savefile()(translate.storage.subtitles.SubStationAlpho	
method), 653	sentence_iter() (translate.lang.fr.fr class method), 370
savefile() (translate.storage.subtitles.SubtitleFile method), 655	sentence_iter() (translate.lang.gu.gu class
savefile() (translate.storage.tbx.tbxfile method), 660	method), 371
savefile() (translate.storage.tiki.TikiStore method),	sentence_iter() (translate.lang.he.he class

method), 371			300	
<pre>sentence_iter() method), 372</pre>	(translate.lang.hi.hi	class	sentencecount() late.filters.checks.L20nChecker	(trans- nethod),
sentence_iter()	(translate.lang.hy.hy	class	306	
<i>method</i>), 373			sentencecount()	(trans-
sentence_iter() method), 374	(translate.lang.ja.ja	class	late.filters.checks.LibreOfficeChecker 1 311	nethod),
sentence_iter()	(translate.lang.km.km	class	sentencecount()	(trans-
method), 375			late.filters.checks.MinimalChecker	nethod),
sentence_iter()	(translate.lang.kn.kn	class	317	
method), 376			sentencecount()	(trans-
sentence_iter()	(translate.lang.ko.ko	class		nethod),
method), 376	` 0		323	,,
sentence_iter()	(translate.lang.ml.ml	class	sentencecount()	(trans-
method), 377	(manustations)	Cress	late.filters.checks.OpenOfficeChecker 1	•
sentence_iter()	(translate.lang.mr.mr	class	328	nemou),
<i>method</i>), 378	(iransiaic.iang.mi.mi	ciass	sentencecount()	(trans-
sentence_iter()	(translate.lang.ne.ne	class		nethod),
	(transtate.tang.ne.ne	ciuss	late.filters.checks.ReducedChecker 1 334	neinoa),
method), 379	(4	-1		(4
sentence_iter()	(translate.lang.pa.pa	class	sentencecount()	(trans-
method), 380			· ·	nethod),
	ranslate.lang.si.si class n	ıethod),	340	
381			sentencecount()	(trans-
sentence_iter()(ta 382	ranslate.lang.st.st class n	nethod),	late.filters.checks.TermChecker 347	nethod),
sentence_iter() method), 382	(translate.lang.sv.sv	class	sentenceend (translate.lang.common.Cattribute), 363	Common
sentence_iter()	(translate.lang.ta.ta	class	sentences () (translate.lang.af.af class metho	d) 357
	(transtate.tang.ta.ta	ciuss		
method), 383	ugualata lawa ta ta alaga w	• • • • • • • • • • • • • • • • • • •	sentences() (translate.lang.am.am class i	neinoa),
	ranslate.lang.te.te class n	ieinoa),		D 250
384	(7	sentences () (translate.lang.ar.ar class metho	
sentence_iter()	(translate.lang.th.th	class	sentences () (translate.lang.bn.bn class meth	
method), 385		_	sentences() (translate.lang.code_or.code_o	or class
sentence_iter()	(translate.lang.ug.ug	class	method), 360	_
method), 386			sentences() (translate.lang.common.Commo	on class
sentence_iter()	(translate.lang.ur.ur	class	method), 363	
method), 386			sentences () (translate.lang.de.de class method	od), 366
sentence_iter()(ta	ranslate.lang.vi.vi class n	ıethod),	sentences () (translate.lang.el.el class method	d), 367
387			sentences () (translate.lang.es.es class metho	od), 367
sentence_iter()	(translate.lang.zh.zh	class	sentences () (translate.lang.fa.fa class metho	(d), 368
method), 388			sentences () (translate.lang.fi.fi class method), 369
sentencecount()		(trans-	sentences () (translate.lang.fr.fr class method	
late.filters.ched	ks.CCLicenseChecker n	`	sentences () (translate.lang.gu.gu class meth	
276		, ,	sentences () (translate.lang.he.he class method	
sentencecount()		(trans-	sentences () (translate.lang.hi.hi class metho	
	ks.DrupalChecker n	nethod),	sentences () (translate.lang.hy.hy class metho	
282	кз. Дириспескет п	ieinou),		
		(4	sentences () (translate.lang.ja.ja class metho	
sentencecount()	1 C CL 1	(trans-	sentences() (translate.lang.km.km class n	петоа),
	ks.GnomeChecker n	ıethod),	375	D 076
288			sentences () (translate.lang.kn.kn class method	
sentencecount()		(trans-	sentences () (translate.lang.ko.ko class metho	
late.filters.chec	ks.IOSChecker method),	294	sentences () (translate.lang.ml.ml class meth	
sentencecount()		(trans-	sentences () (translate.lang.mr.mr class meth	od), 378
late.filters.chec	ks.KdeChecker n	ıethod),	sentences () (translate.lang.ne.ne class method	od), 379

sentences () (translate.lang.pa.pa class method), 380	serialize() (trans-
sentences () (translate.lang.si.si class method), 381	late.storage.omegat.OmegaTFileTab method),
sentences () (translate.lang.st.st class method), 382	502
sentences () (translate.lang.sv.sv class method), 382	serialize() (translate.storage.oo.oofile method),
sentences () (translate.lang.ta.ta class method), 383	506
sentences () (translate.lang.te.te class method), 384	serialize() (translate.storage.php.LaravelPHPFile
sentences () (translate.lang.th.th class method), 385	method), 559
sentences () (translate.lang.ug.ug class method), 386	<pre>serialize() (translate.storage.php.phpfile method),</pre>
sentences () (translate.lang.ur.ur class method), 387	564
sentences () (translate.lang.vi.vi class method), 387	serialize() (translate.storage.pocommon.pofile
sentences () (translate.lang.zh.zh class method), 388	method), 570
serialize() (translate.storage.base.DictStore	serialize() (translate.storage.poxliff.PoXliffFile
method), 405	method), 579
serialize() (translate.storage.base.TranslationStore	serialize() (translate.storage.properties.gwtfile
method), 410	method), 597
serialize() (translate.storage.catkeys.CatkeysFile	serialize() (translate.storage.properties.javafile
method), 417	method), 599
serialize() (translate.storage.csvl10n.csvfile	serialize() (trans-
method), 422	late.storage.properties.javautf16file method),
serialize() (translate.storage.dtd.dtdfile method),	601
428	serialize() (translate.storage.properties.javautf8file
serialize() (translate.storage.html.htmlfile method),	method), 603
437	serialize() (translate.storage.properties.joomlafile
serialize() (translate.storage.html.POHTMLParser	method), 604
method), 435	serialize() (translate.storage.properties.propfile
serialize() (translate.storage.ical.icalfile method),	method), 606
443	serialize() (translate.storage.properties.stringsfile
443	Serrarrze () (transiale.siorage.properties.stringsfile
serialize() (translate.storage.ini.inifile method),	method), 614
448	<pre>method), 614 serialize() (trans-</pre>
	method), 614
448	<pre>method), 614 serialize() (trans-</pre>
448 serialize() (trans-	method), 614 serialize() (trans- late.storage.properties.stringsutf8file method),
448 serialize() (trans- late.storage.jsonl10n.ARBJsonFile method),	method), 614 serialize() (trans- late.storage.properties.stringsutf8file method), 616 serialize() (translate.storage.properties.xwikifile
448 serialize() (trans- late.storage.jsonl10n.ARBJsonFile method), 453 serialize() (trans-	method), 614 serialize() (trans- late.storage.properties.stringsutf8file method), 616 serialize() (translate.storage.properties.xwikifile method), 618
448 serialize() (trans- late.storage.jsonl10n.ARBJsonFile method), 453 serialize() (trans- late.storage.jsonl10n.Gol18NJsonFile	method), 614 serialize() (trans- late.storage.properties.stringsutf8file method), 616 serialize() (translate.storage.properties.xwikifile method), 618 serialize() (translate.storage.properties.xwikifile)
448 serialize() (trans- late.storage.jsonl10n.ARBJsonFile method), 453 serialize() (trans- late.storage.jsonl10n.Gol18NJsonFile method), 458	method), 614 serialize() (trans- late.storage.properties.stringsutf8file method), 616 serialize() (translate.storage.properties.xwikifile method), 618 serialize() (translate.storage.properties.xwikifule) late.storage.properties.XWikiFullPage
448 serialize() (trans- late.storage.jsonl10n.ARBJsonFile method), 453 serialize() (trans- late.storage.jsonl10n.Go118NJsonFile method), 458 serialize() (translate.storage.jsonl10n.I18NextFile	method), 614 serialize() (trans- late.storage.properties.stringsutf8file method), 616 serialize() (translate.storage.properties.xwikifile method), 618 serialize() (translate.storage.properties.xwikifile method), 593 (trans- late.storage.properties.XWikiFullPage method), 593
448 serialize() (trans- late.storage.jsonl10n.ARBJsonFile method), 453 serialize() (trans- late.storage.jsonl10n.GoI18NJsonFile method), 458 serialize() (translate.storage.jsonl10n.I18NextFile method), 463	method), 614 serialize() (trans- late.storage.properties.stringsutf8file method), 616 serialize() (translate.storage.properties.xwikifile method), 618 serialize() (trans- late.storage.properties.XWikiFullPage method), 593 serialize() (trans-
serialize() (trans- late.storage.jsonl10n.ARBJsonFile method), 453 serialize() (trans- late.storage.jsonl10n.GoI18NJsonFile method), 458 serialize() (translate.storage.jsonl10n.I18NextFile method), 463 serialize() (translate.storage.jsonl10n.JsonFile	method), 614 serialize() (trans- late.storage.properties.stringsutf8file method), 616 serialize() (translate.storage.properties.xwikifile method), 618 serialize() (trans- late.storage.properties.XWikiFullPage method), 593 serialize() (trans- late.storage.properties.XWikiPageProperties
448 serialize() (trans- late.storage.jsonl10n.ARBJsonFile method), 453 serialize() (trans- late.storage.jsonl10n.GoI18NJsonFile method), 458 serialize() (translate.storage.jsonl10n.I18NextFile method), 463	method), 614 serialize() (trans- late.storage.properties.stringsutf8file method), 616 serialize() (translate.storage.properties.xwikifile method), 618 serialize() (trans- late.storage.properties.XWikiFullPage method), 593 serialize() (trans-
serialize() (trans- late.storage.jsonl10n.ARBJsonFile method), 453 serialize() (trans- late.storage.jsonl10n.GoI18NJsonFile method), 458 serialize() (translate.storage.jsonl10n.I18NextFile method), 463 serialize() (translate.storage.jsonl10n.JsonFile	method), 614 serialize() (trans- late.storage.properties.stringsutf8file method), 616 serialize() (translate.storage.properties.xwikifile method), 618 serialize() (trans- late.storage.properties.XWikiFullPage method), 593 serialize() (trans- late.storage.properties.XWikiPageProperties
serialize() (trans- late.storage.jsonl10n.ARBJsonFile method), 453 serialize() (trans- late.storage.jsonl10n.GoI18NJsonFile method), 458 serialize() (translate.storage.jsonl10n.I18NextFile method), 463 serialize() (translate.storage.jsonl10n.JsonFile method), 468 serialize() (translate.storage.jsonl10n.JsonFile	method), 614 serialize() (trans- late.storage.properties.stringsutf8file method), 616 serialize() (translate.storage.properties.xwikifile method), 618 serialize() (trans- late.storage.properties.XWikiFullPage method), 593 serialize() (trans- late.storage.properties.XWikiPageProperties method), 595
serialize() (trans- late.storage.jsonl10n.ARBJsonFile method), 453 serialize() (trans- late.storage.jsonl10n.GoI18NJsonFile method), 458 serialize() (translate.storage.jsonl10n.I18NextFile method), 463 serialize() (translate.storage.jsonl10n.JsonFile method), 468 serialize() (translate.storage.jsonl10n.JsonFile method), 468 serialize() (translate.storage.jsonl10n.JsonFile method), 468	method), 614 serialize() (trans- late.storage.properties.stringsutf8file method), 616 serialize() (translate.storage.properties.xwikifile method), 618 serialize() (trans- late.storage.properties.XWikiFullPage method), 593 serialize() (trans- late.storage.properties.XWikiPageProperties method), 595 serialize() (translate.storage.pypo.pofile method), 624
serialize() (trans- late.storage.jsonl10n.ARBJsonFile method), 453 serialize() (trans- late.storage.jsonl10n.GoI18NJsonFile method), 458 serialize() (translate.storage.jsonl10n.I18NextFile method), 463 serialize() (translate.storage.jsonl10n.JsonFile method), 468 serialize() (translate.storage.jsonl10n.JsonFile method), 468 serialize() (translate.storage.jsonl10n.JsonFile method), 468	method), 614 serialize() (trans- late.storage.properties.stringsutf8file method), 616 serialize() (translate.storage.properties.xwikifile method), 618 serialize() (translate.storage.properties.XWikiFullPage method), 593 serialize() (translate.storage.properties method), 595 serialize() (translate.storage.pypo.pofile method), 624 serialize() (translate.storage.qm.qmfile method),
serialize() (trans- late.storage.jsonl10n.ARBJsonFile method), 453 serialize() (trans- late.storage.jsonl10n.GoI18NJsonFile method), 458 serialize() (translate.storage.jsonl10n.I18NextFile method), 463 serialize() (translate.storage.jsonl10n.JsonFile method), 468 serialize() (translate.storage.jsonl10n.JsonFile method), 468 serialize() (translate.storage.jsonl10n.JsonFile method), 468 serialize() (trans- late.storage.jsonl10n.JsonNestedFile method), 470 serialize() (trans-	method), 614 serialize() (trans- late.storage.properties.stringsutf8file method), 616 serialize() (translate.storage.properties.xwikifile method), 618 serialize() (translate.storage.properties.XWikiFullPage method), 593 serialize() (trans- late.storage.properties.XWikiPageProperties method), 595 serialize() (translate.storage.pypo.pofile method), 624 serialize() (translate.storage.qm.qmfile method), 630
448 serialize() (translate.storage.jsonl10n.ARBJsonFile method), 453 serialize() (translate.storage.jsonl10n.Go118NJsonFile method), 458 serialize() (translate.storage.jsonl10n.I18NextFile method), 463 serialize() (translate.storage.jsonl10n.JsonFile method), 468 serialize() (translate.storage.jsonl10n.JsonFile method), 468 serialize() (translate.storage.jsonl10n.JsonNestedFile method), 470 serialize() (translate.storage.jsonl10n.WebExtensionJsonFile	method), 614 serialize() (trans- late.storage.properties.stringsutf8file method), 616 serialize() (translate.storage.properties.xwikifile method), 618 serialize() (trans- late.storage.properties.XWikiFullPage method), 593 serialize() (trans- late.storage.properties.XWikiPageProperties method), 595 serialize() (translate.storage.pypo.pofile method), 624 serialize() (translate.storage.qm.qmfile method), 630 serialize() (translate.storage.qph.QphFile method),
serialize() (translate.storage.jsonl10n.ARBJsonFile method), 453 serialize() (translate.storage.jsonl10n.Go118NJsonFile method), 458 serialize() (translate.storage.jsonl10n.I18NextFile method), 463 serialize() (translate.storage.jsonl10n.JsonFile method), 468 serialize() (translate.storage.jsonl10n.JsonFile method), 468 serialize() (translate.storage.jsonl10n.JsonNestedFile method), 470 serialize() (translate.storage.jsonl10n.WebExtensionJsonFile method), 478	method), 614 serialize() (trans- late.storage.properties.stringsutf8file method), 616 serialize() (translate.storage.properties.xwikifile method), 618 serialize() (translate.storage.properties.XWikiFullPage method), 593 serialize() (translate.storage.properties method), 595 serialize() (translate.storage.pypo.pofile method), 624 serialize() (translate.storage.qm.qmfile method), 630 serialize() (translate.storage.qph.QphFile method), 635
serialize() (trans- late.storage.jsonl10n.ARBJsonFile method), 453 serialize() (trans- late.storage.jsonl10n.Go118NJsonFile method), 458 serialize() (translate.storage.jsonl10n.I18NextFile method), 463 serialize() (translate.storage.jsonl10n.JsonFile method), 468 serialize() (translate.storage.jsonl10n.JsonFile method), 468 serialize() (trans- late.storage.jsonl10n.JsonNestedFile method), 470 serialize() (trans- late.storage.jsonl10n.WebExtensionJsonFile method), 478 serialize() (translate.storage.lisa.LISAfile method),	method), 614 serialize() (trans- late.storage.properties.stringsutf8file method), 616 serialize() (translate.storage.properties.xwikifile method), 618 serialize() (translate.storage.properties.XWikiFullPage method), 593 serialize() (translate.storage.properties method), 595 serialize() (translate.storage.pypo.pofile method), 624 serialize() (translate.storage.qm.qmfile method), 630 serialize() (translate.storage.qph.QphFile method), 635 serialize() (translate.storage.rc.rcfile method), 641
serialize() (translate.storage.jsonl10n.ARBJsonFile method), 453 serialize() (translate.storage.jsonl10n.Go118NJsonFile method), 458 serialize() (translate.storage.jsonl10n.I18NextFile method), 463 serialize() (translate.storage.jsonl10n.JsonFile method), 468 serialize() (translate.storage.jsonl10n.JsonFile method), 468 serialize() (translate.storage.jsonl10n.JsonNestedFile method), 470 serialize() (translate.storage.jsonl10n.WebExtensionJsonFile method), 478	method), 614 serialize() (trans- late.storage.properties.stringsutf8file method), 616 serialize() (translate.storage.properties.xwikifile method), 618 serialize() (translate.storage.properties.XWikiFullPage method), 593 serialize() (trans- late.storage.properties.XWikiPageProperties method), 595 serialize() (translate.storage.pypo.pofile method), 624 serialize() (translate.storage.qm.qmfile method), 630 serialize() (translate.storage.qph.QphFile method), 635 serialize() (translate.storage.rc.rcfile method), 641 serialize() (translate.storage.rc.rcfile method), 641
serialize() (trans- late.storage.jsonl10n.ARBJsonFile method), 453 serialize() (trans- late.storage.jsonl10n.Go118NJsonFile method), 458 serialize() (translate.storage.jsonl10n.I18NextFile method), 463 serialize() (translate.storage.jsonl10n.JsonFile method), 468 serialize() (translate.storage.jsonl10n.JsonFile method), 468 serialize() (trans- late.storage.jsonl10n.JsonNestedFile method), 470 serialize() (trans- late.storage.jsonl10n.WebExtensionJsonFile method), 478 serialize() (translate.storage.lisa.LISAfile method),	method), 614 serialize() (trans- late.storage.properties.stringsutf8file method), 616 serialize() (translate.storage.properties.xwikifile method), 618 serialize() (translate.storage.properties.XWikiFullPage method), 593 serialize() (translate.storage.properties method), 595 serialize() (translate.storage.pypo.pofile method), 624 serialize() (translate.storage.qm.qmfile method), 630 serialize() (translate.storage.qph.QphFile method), 635 serialize() (translate.storage.rc.rcfile method), 641
serialize() (trans- late.storage.jsonl10n.ARBJsonFile method), 453 serialize() (trans- late.storage.jsonl10n.Go118NJsonFile method), 458 serialize() (translate.storage.jsonl10n.I18NextFile method), 463 serialize() (translate.storage.jsonl10n.JsonFile method), 468 serialize() (translate.storage.jsonl10n.JsonFile method), 468 serialize() (trans- late.storage.jsonl10n.JsonNestedFile method), 470 serialize() (trans- late.storage.jsonl10n.WebExtensionJsonFile method), 478 serialize() (translate.storage.lisa.LISAfile method), 483	method), 614 serialize() (trans- late.storage.properties.stringsutf8file method), 616 serialize() (translate.storage.properties.xwikifile method), 618 serialize() (trans- late.storage.properties.XWikiFullPage method), 593 serialize() (trans- late.storage.properties.XWikiPageProperties method), 595 serialize() (translate.storage.pypo.pofile method), 624 serialize() (translate.storage.qm.qmfile method), 630 serialize() (translate.storage.qph.QphFile method), 635 serialize() (translate.storage.rc.rcfile method), 641 serialize() (translate.storage.rc.rcfile method), 641 serialize() (translate.storage.subtitles.AdvSubStationAlphaFile
serialize() (trans- late.storage.jsonl10n.ARBJsonFile method), 453 serialize() (trans- late.storage.jsonl10n.Go118NJsonFile method), 458 serialize() (translate.storage.jsonl10n.I18NextFile method), 463 serialize() (translate.storage.jsonl10n.JsonFile method), 468 serialize() (translate.storage.jsonl10n.JsonFile method), 468 serialize() (translate.storage.jsonl10n.JsonNestedFile method), 470 serialize() (translate.storage.jsonl10n.WebExtensionJsonFile method), 478 serialize() (translate.storage.lisa.LISAfile method), 483 serialize() (translate.storage.mo.mofile method), 489	method), 614 serialize() (trans- late.storage.properties.stringsutf8file method), 616 serialize() (translate.storage.properties.xwikifile method), 618 serialize() (trans- late.storage.properties.XWikiFullPage method), 593 serialize() (trans- late.storage.properties.XWikiPageProperties method), 595 serialize() (translate.storage.pypo.pofile method), 624 serialize() (translate.storage.qm.qmfile method), 630 serialize() (translate.storage.qph.QphFile method), 635 serialize() (translate.storage.rc.rcfile method), 641 serialize() (translate.storage.subtitles.AdvSubStationAlphaFile method), 648
serialize() (trans- late.storage.jsonl10n.ARBJsonFile method), 453 serialize() (trans- late.storage.jsonl10n.Go118NJsonFile method), 458 serialize() (translate.storage.jsonl10n.I18NextFile method), 463 serialize() (translate.storage.jsonl10n.JsonFile method), 468 serialize() (translate.storage.jsonl10n.JsonFile method), 468 serialize() (translate.storage.jsonl10n.JsonNestedFile method), 470 serialize() (translate.storage.jsonl10n.WebExtensionJsonFile method), 478 serialize() (translate.storage.lisa.LISAfile method), 483 serialize() (translate.storage.mo.mofile method), 489 serialize() (translate.storage.mo.mofile method),	method), 614 serialize() (trans- late.storage.properties.stringsutf8file method), 616 serialize() (translate.storage.properties.xwikifile method), 618 serialize() (trans- late.storage.properties.XWikiFullPage method), 593 serialize() (trans- late.storage.properties.XWikiPageProperties method), 595 serialize() (translate.storage.pypo.pofile method), 624 serialize() (translate.storage.qm.qmfile method), 630 serialize() (translate.storage.qph.QphFile method), 635 serialize() (translate.storage.rc.rcfile method), 641 serialize() (translate.storage.subtitles.AdvSubStationAlphaFile method), 648 serialize() (trans-
serialize() (trans- late.storage.jsonl10n.ARBJsonFile method), 453 serialize() (trans- late.storage.jsonl10n.Go118NJsonFile method), 458 serialize() (translate.storage.jsonl10n.I18NextFile method), 463 serialize() (translate.storage.jsonl10n.JsonFile method), 468 serialize() (translate.storage.jsonl10n.JsonFile method), 468 serialize() (trans- late.storage.jsonl10n.JsonNestedFile method), 470 serialize() (translate.storage.lisa.LISAfile method), 483 serialize() (translate.storage.mo.mofile method), 489 serialize() (translate.storage.mo.mofile method), late.storage.mozilla_lang.LangStore method),	method), 614 serialize() (trans- late.storage.properties.stringsutf8file method), 616 serialize() (translate.storage.properties.xwikifile method), 618 serialize() (translate.storage.properties.xWikiFullPage method), 593 serialize() (trans- late.storage.properties.xWikiPageProperties method), 595 serialize() (translate.storage.pypo.pofile method), 624 serialize() (translate.storage.qm.qmfile method), 630 serialize() (translate.storage.qph.QphFile method), 635 serialize() (translate.storage.rc.rcfile method), 641 serialize() (translate.storage.rc.rcfile method), 641 serialize() (translate.storage.subtitles.AdvSubStationAlphaFile method), 648 serialize() (translate.storage.subtitles.MicroDVDFile method), late.storage.subtitles.MicroDVDFile method),
serialize() (translate.storage.jsonl10n.ARBJsonFile method), 453 serialize() (translate.storage.jsonl10n.Go118NJsonFile method), 458 serialize() (translate.storage.jsonl10n.I18NextFile method), 463 serialize() (translate.storage.jsonl10n.JsonFile method), 468 serialize() (translate.storage.jsonl10n.JsonFile method), 470 serialize() (translate.storage.jsonl10n.WebExtensionJsonFile method), 478 serialize() (translate.storage.lisa.LISAfile method), 483 serialize() (translate.storage.mo.mofile method), 489 serialize() (translate.storage.mo.mofile method), 489 serialize() (translate.storage.mo.mofile method), 495	method), 614 serialize() (trans- late.storage.properties.stringsutf8file method), 616 serialize() (translate.storage.properties.xwikifile method), 618 serialize() (trans- late.storage.properties.XWikiFullPage method), 593 serialize() (trans- late.storage.properties.XWikiPageProperties method), 595 serialize() (translate.storage.pypo.pofile method), 624 serialize() (translate.storage.qm.qmfile method), 630 serialize() (translate.storage.qph.QphFile method), 635 serialize() (translate.storage.rc.rcfile method), 641 serialize() (translate.storage.rc.rcfile method), 641 serialize() (translate.storage.subtitles.AdvSubStationAlphaFile method), 648 serialize() (translate.storage.subtitles.MicroDVDFile method), 650
serialize() (trans- late.storage.jsonl10n.ARBJsonFile method), 453 serialize() (trans- late.storage.jsonl10n.Go118NJsonFile method), 458 serialize() (translate.storage.jsonl10n.I18NextFile method), 463 serialize() (translate.storage.jsonl10n.JsonFile method), 468 serialize() (translate.storage.jsonl10n.JsonFile method), 468 serialize() (trans- late.storage.jsonl10n.JsonNestedFile method), 470 serialize() (translate.storage.lisa.LISAfile method), 483 serialize() (translate.storage.mo.mofile method), 489 serialize() (translate.storage.mo.mofile method), late.storage.mozilla_lang.LangStore method),	method), 614 serialize() (trans- late.storage.properties.stringsutf8file method), 616 serialize() (translate.storage.properties.xwikifile method), 618 serialize() (translate.storage.properties.xWikiFullPage method), 593 serialize() (trans- late.storage.properties.xWikiPageProperties method), 595 serialize() (translate.storage.pypo.pofile method), 624 serialize() (translate.storage.qm.qmfile method), 630 serialize() (translate.storage.qph.QphFile method), 635 serialize() (translate.storage.rc.rcfile method), 641 serialize() (translate.storage.rc.rcfile method), 641 serialize() (translate.storage.subtitles.AdvSubStationAlphaFile method), 648 serialize() (translate.storage.subtitles.MicroDVDFile method), late.storage.subtitles.MicroDVDFile method),

serialize() (trans	1
late.storage.subtitles.SubStationAlphaFile	method), 396
method), 653	set_usage() (trans-
serialize() (translate.storage.subtitles.SubtitleFil method), 655	e late.tools.poconflicts.ConflictOptionParser method), 717
serialize() (translate.storage.tbx.tbxfile method)	
660	late.tools.pogrep.GrepOptionParser method),
serialize() (translate.storage.tiki.TikiStore method)	
666	set_usage() (trans-
serialize() (translate.storage.tmx.tmxfile method)), late.tools.porestructure.SplitOptionParser method), 725
serialize() (trans	_ , , , , , , , , , , , , , , , , , , ,
late.storage.trados.TradosTxtTmFile method/ 680), late.tools.poterminology.TerminologyOptionParser method), 728
serialize() (translate.storage.ts2.tsfile method), 68	
serialize() (translate.storage.txt.TxtFile method) 688), late.storage.catkeys.CatkeysHeader method), 417
<pre>serialize() (translate.storage.utx.UtxFile method)</pre>), setconfig() (trans-
693	late.filters.checks.CCLicenseChecker method),
serialize() (trans	
late.storage.wordfast.WordfastTMFile method 699), setconfig() (translate.filters.checks.DrupalChecker method), 282
serialize() (translate.storage.xliff.xlifffile method) 706), setconfig() (translate.filters.checks.GnomeChecker method), 288
SeriousFilterFailure, 336	setconfig() (translate.filters.checks.IOSChecker
$set_time()$ (translate.storage.trados.TradosTxtDat	e method), 294
method), 675	setconfig() (translate.filters.checks.KdeChecker
<pre>set_time() (translate.storage.wordfast.WordfastTim</pre>	
method), 700	setconfig() (translate.filters.checks.L20nChecker
set_timestring() (trans	
late.storage.trados.TradosTxtDate method	
675	late.filters.checks.LibreOfficeChecker method),
set_timestring() (trans	
late.storage.wordfast.WordfastTime method 700	
	late.filters.checks.MinimalChecker method), 317
_ , , ,	ser setconfig () (translate.filters.checks.MozillaChecker
method), 240	method), 323
set_usage() (trans	
late.convert.convert.ConvertOptionParser	late.filters.checks.OpenOfficeChecker method),
method), 243	329
set_usage() (trans	setconfig() (trans-
late.convert.po2moz.MozConvertOptionParser	
method), 255	334
set_usage() (trans	- setconfig() (trans-
late.convert.po2tmx.TmxOptionParser	late.filters.checks.StandardChecker method),
method), 261	340
set_usage() (trans	\$
late.convert.po2wordfast.WfOptionParser method), 265	late.filters.checks.StandardUnitChecker method), 343
set_usage() (trans	
late.filters.pofilter.FilterOptionParser method	
355	setconfig() (trans-
set_usage() (trans	late.filters.checks.TranslationChecker method),

350	setcontext() (translate.storage.poxliff.PoXliffUnit
setconfig() (translate.filters.checks.UnitChecker	method), 583
method), 350	setcontext() (trans-
<pre>setcontext() (translate.storage.base.DictUnit</pre>	late.storage.properties.proppluralunit method),
method), 408	609
setcontext() (trans-	<pre>setcontext() (translate.storage.properties.propunit</pre>
late.storage.base.TranslationUnit method),	method), 612
413	setcontext() (translate.storage.properties.xwikiunit
<pre>setcontext() (translate.storage.catkeys.CatkeysUnit</pre>	method), 621
method), 420	setcontext() (translate.storage.pypo.pounit
setcontext() (translate.storage.csvl10n.csvunit	method), 627
method), 425	setcontext() (translate.storage.qm.qmunit method),
setcontext() (translate.storage.dtd.dtdunit method),	633
431	setcontext() (translate.storage.qph.QphUnit
setcontext() (translate.storage.html.htmlunit	method), 639
method), 440	setcontext() (translate.storage.rc.rcunit method),
	644
setcontext() (translate.storage.ical.icalunit	
method), 446	setcontext() (trans-
setcontext() (translate.storage.ini.iniunit method), 451	late.storage.subtitles.SubtitleUnit method), 658
setcontext() (trans-	<pre>setcontext() (translate.storage.tbx.tbxunit method),</pre>
late.storage.jsonl10n.ARBJsonUnit method),	664
457	setcontext() (translate.storage.tiki.TikiUnit
setcontext() (trans-	method), 669
late.storage.jsonl10n.GoI18NJsonUnit	setcontext() (translate.storage.tmx.tmxunit
method), 462	method), 674
setcontext() (trans-	<pre>setcontext() (translate.storage.trados.TradosUnit</pre>
late.storage.jsonl10n.I18NextUnit method),	method), 678
467	<pre>setcontext() (translate.storage.ts2.tsunit method),</pre>
setcontext() (trans-	686
late.storage.jsonl10n.JsonNestedUnit method),	setcontext() (translate.storage.txt.TxtUnit method),
473	691
setcontext() (translate.storage.jsonl10n.JsonUnit	setcontext() (translate.storage.utx.UtxUnit
method), 476	method), 697
setcontext() (trans-	setcontext() (trans-
late.storage.jsonl10n.WebExtensionJsonUnit	late.storage.wordfast.WordfastUnit method),
method), 481	703
setcontext() (translate.storage.lisa.LISAunit	
method), 487	method), 710
	setdefault() (translate.misc.dictutils.cidict
setcontext() (translate.storage.mo.mounit method), 493	method), 389
setcontext() (trans-	setdefault() (translate.storage.oo.unormalizechar
late.storage.mozilla_lang.LangUnit method),	method), 507
498	setdict() (translate.storage.catkeys.CatkeysUnit
setcontext() (trans-	method), 420
late.storage.omegat.OmegaTUnit method), 505	setdict() (translate.storage.omegat.OmegaTUnit method), 505
setcontext() (trans-	<pre>setdict() (translate.storage.utx.UtxUnit method),</pre>
late.storage.php.LaravelPHPUnit method),	697
562	<pre>setdict() (translate.storage.wordfast.WordfastUnit</pre>
setcontext() (translate.storage.php.phpunit	method), 703
method), 567	seterrorleveloptions() (trans-
<pre>setcontext() (translate.storage.pocommon.pounit</pre>	late.convert.convert.ArchiveConvertOptionParser
method), 573	method), 240

seterrorleveloptions() (tran	s- method), 396
late.convert.convert.ConvertOptionParser	setformats() (trans-
method), 243	late. tools. poconflicts. Conflict Option Parser
seterrorleveloptions() (tran	s- <i>method</i>), 718
late.convert.po2moz.MozConvertOptionParse	r setformats() (trans-
method), 255	late.tools.pogrep.GrepOptionParser method),
seterrorleveloptions() (tran	
late.convert.po2tmx.TmxOptionParser	setformats() (trans-
method), 261	late.tools.porestructure.SplitOptionParser
seterrorleveloptions() (tran	
late.convert.po2wordfast.WfOptionParser	setformats() (trans-
method), 265	late.tools.poterminology.TerminologyOptionParser
seterrorleveloptions() (tran	
late.filters.pofilter.FilterOptionParser method	
355	setid() (translate.storage.base.TranslationUnit
seterrorleveloptions() (tran	
late.misc.optrecurse.RecursiveOptionParser	setid() (translate.storage.catkeys.CatkeysUnit
method), 396	method), 420
seterrorleveloptions() (tran	
late.tools.poconflicts.ConflictOptionParser	425
method), 717	setid() (translate.storage.dtd.dtdunit method), 431
seterrorleveloptions() (tran	
late.tools.pogrep.GrepOptionParser method	
721	setid() (translate.storage.ini.iniunit method), 451
seterrorleveloptions() (tran late.tools.porestructure.SplitOptionParser	method), 457
method), 725	setid() (translate.storage.jsonl10n.GoI18NJsonUnit
meinoa), 723	
seterrorleveloptions() (tran	s- method), 462
seterrorleveloptions() (translate.tools.poterminology.TerminologyOptionF	s- method), 462 arsesetid() (translate.storage.jsonl10n.I18NextUnit
seterrorleveloptions() (tran late.tools.poterminology.TerminologyOptionF method), 728	s- method), 462 arsesetid() (translate.storage.jsonl10n.I18NextUnit method), 467
seterrorleveloptions() (translate.tools.poterminology.TerminologyOptionFounthod), 728 setfilename() (translate.storage.poxliff.PoXliffFi	s- method), 462 arsebetid() (translate.storage.jsonl10n.118NextUnit method), 467 le setid() (translate.storage.jsonl10n.JsonNestedUnit
seterrorleveloptions() (translate.tools.poterminology.TerminologyOptionFoundthod), 728 setfilename() (translate.storage.poxliff.PoXliffFilemethod), 579	s- method), 462 arsebetid() (translate.storage.jsonl10n.118NextUnit method), 467 le setid() (translate.storage.jsonl10n.JsonNestedUnit method), 473
seterrorleveloptions() (translate.tools.poterminology.TerminologyOptionFinethod), 728 setfilename() (translate.storage.poxliff.PoXliffFinethod), 579 setfilename() (translate.storage.xliff.xlifffinethod)	method), 462 arsebetid() (translate.storage.jsonl10n.118NextUnit method), 467 le setid() (translate.storage.jsonl10n.JsonNestedUnit method), 473 le setid() (translate.storage.jsonl10n.JsonUnit method),
seterrorleveloptions() (translate.tools.poterminology.TerminologyOptionFormethod), 728 setfilename() (translate.storage.poxliff.PoXliffFirmethod), 579 setfilename() (translate.storage.xliff.xlifffirmethod), 707	method), 462 arsebetid() (translate.storage.jsonl10n.I18NextUnit method), 467 le setid() (translate.storage.jsonl10n.JsonNestedUnit method), 473 le setid() (translate.storage.jsonl10n.JsonUnit method), 476
seterrorleveloptions() (translate.tools.poterminology.TerminologyOptionFormethod), 728 setfilename() (translate.storage.poxliff.PoXliffFirmethod), 579 setfilename() (translate.storage.xliff.xlifffirmethod), 707 setformats() (translate.storage.xliff.xlifffirmethod)	method), 462 arsebetid() (translate.storage.jsonl10n.I18NextUnit method), 467 le setid() (translate.storage.jsonl10n.JsonNestedUnit method), 473 le setid() (translate.storage.jsonl10n.JsonUnit method), 476 setid() (translate.storage.jsonl10n.WebExtensionJsonUnit
seterrorleveloptions() (translate.tools.poterminology.TerminologyOptionFormethod), 728 setfilename() (translate.storage.poxliff.PoXliffFirmethod), 579 setfilename() (translate.storage.xliff.xlifffirmethod), 707 setformats() (translate.convert.ArchiveConvertOptionPage.	method), 462 arsebetid() (translate.storage.jsonl10n.118NextUnit method), 467 le setid() (translate.storage.jsonl10n.JsonNestedUnit method), 473 le setid() (translate.storage.jsonl10n.JsonUnit method), 476 s- setid() (translate.storage.jsonl10n.WebExtensionJsonUnit method), 481
seterrorleveloptions() (translate.tools.poterminology.TerminologyOptionFormethod), 728 setfilename() (translate.storage.poxliff.PoXliffFirmethod), 579 setfilename() (translate.storage.xliff.xlifffirmethod), 707 setformats() (translate.convert.ArchiveConvertOptionPainethod), 240	method), 462 arsebetid() (translate.storage.jsonl10n.118NextUnit method), 467 le setid() (translate.storage.jsonl10n.JsonNestedUnit method), 473 le setid() (translate.storage.jsonl10n.JsonUnit method), 476 s- setid() (translate.storage.jsonl10n.WebExtensionJsonUnit reser method), 481 setid() (translate.storage.lisa.LISAunit method), 487
seterrorleveloptions() (translate.tools.poterminology.TerminologyOptionFormethod), 728 setfilename() (translate.storage.poxliff.PoXliffFirmethod), 579 setfilename() (translate.storage.xliff.xlifffirmethod), 707 setformats() (translate.convert.ArchiveConvertOptionParamethod), 240 setformats() (translate.storage.xliff.xliffirmethod), 240	method), 462 arsebetid() (translate.storage.jsonl10n.118NextUnit method), 467 le setid() (translate.storage.jsonl10n.JsonNestedUnit method), 473 le setid() (translate.storage.jsonl10n.JsonUnit method), 476 s- setid() (translate.storage.jsonl10n.WebExtensionJsonUnit reser method), 481 setid() (translate.storage.lisa.LISAunit method), 487 s- setid() (translate.storage.mo.mounit method), 493
seterrorleveloptions() (translate.tools.poterminology.TerminologyOptionFormethod), 728 setfilename() (translate.storage.poxliff.PoXliffFirmethod), 579 setfilename() (translate.storage.xliff.xlifffirmethod), 707 setformats() (translate.convert.ArchiveConvertOptionParmethod), 240 setformats() (translate.convert.ConvertOptionParser	method), 462 arsebetid() (translate.storage.jsonl10n.118NextUnit method), 467 le setid() (translate.storage.jsonl10n.JsonNestedUnit method), 473 le setid() (translate.storage.jsonl10n.JsonUnit method), 476 s- setid() (translate.storage.jsonl10n.WebExtensionJsonUnit reser method), 481 setid() (translate.storage.lisa.LISAunit method), 487 s- setid() (translate.storage.mo.mounit method), 493 setid() (translate.storage.mo.mounit method), 493 setid() (translate.storage.mozilla_lang.LangUnit
seterrorleveloptions() (translate.tools.poterminology.TerminologyOptionFormethod), 728 setfilename() (translate.storage.poxliff.PoXliffFirmethod), 579 setfilename() (translate.storage.xliff.xlifffirmethod), 707 setformats() (translate.convert.ArchiveConvertOptionParmethod), 240 setformats() (translate.convert.Convert.ConvertOptionParsermethod), 243	method), 462 arsebetid() (translate.storage.jsonl10n.I18NextUnit method), 467 le setid() (translate.storage.jsonl10n.JsonNestedUnit method), 473 le setid() (translate.storage.jsonl10n.JsonUnit method), 476 s- setid() (translate.storage.jsonl10n.WebExtensionJsonUnit reser method), 481 setid() (translate.storage.lisa.LISAunit method), 487 s- setid() (translate.storage.mo.mounit method), 493 setid() (translate.storage.mozilla_lang.LangUnit method), 498
seterrorleveloptions() (translate.tools.poterminology.TerminologyOptionFormethod), 728 setfilename() (translate.storage.poxliff.PoXliffFirmethod), 579 setfilename() (translate.storage.xliff.xlifffirmethod), 707 setformats() (translate.convert.ArchiveConvertOptionParamethod), 240 setformats() (translate.convert.convert.ConvertOptionParamethod), 243 setformats() (translate.convert.convert.ConvertOptionParamethod), 243	method), 462 arsebetid() (translate.storage.jsonl10n.118NextUnit method), 467 le setid() (translate.storage.jsonl10n.JsonNestedUnit method), 473 le setid() (translate.storage.jsonl10n.JsonUnit method), 476 s- setid() (translate.storage.jsonl10n.WebExtensionJsonUnit method), 481 setid() (translate.storage.lisa.LISAunit method), 487 s- setid() (translate.storage.mo.mounit method), 493 setid() (translate.storage.mozilla_lang.LangUnit method), 498 s- setid() (translate.storage.omegat.OmegaTUnit
seterrorleveloptions() (translate.tools.poterminology.TerminologyOptionFormethod), 728 setfilename() (translate.storage.poxliff.PoXliffFirmethod), 579 setfilename() (translate.storage.xliff.xlifffirmethod), 707 setformats() (translate.convert.ArchiveConvertOptionParamethod), 240 setformats() (translate.convert.ConvertOptionParamethod), 243 setformats() (translate.convert.convert.ConvertOptionParamethod), 243 setformats() (translate.convert.po2moz.MozConvertOptionParamethod)	method), 462 arsebetid() (translate.storage.jsonl10n.118NextUnit method), 467 le setid() (translate.storage.jsonl10n.JsonNestedUnit method), 473 le setid() (translate.storage.jsonl10n.JsonUnit method), 476 s- setid() (translate.storage.jsonl10n.WebExtensionJsonUnit method), 481 setid() (translate.storage.lisa.LISAunit method), 487 s- setid() (translate.storage.mo.mounit method), 493 setid() (translate.storage.mozilla_lang.LangUnit method), 498 s- setid() (translate.storage.omegat.OmegaTUnit method), 505
seterrorleveloptions() (translate.tools.poterminology.TerminologyOptionFormethod), 728 setfilename() (translate.storage.poxliff.PoXliffFirmethod), 579 setfilename() (translate.storage.xliff.xlifffirmethod), 707 setformats() (translate.convert.ArchiveConvertOptionParamethod), 240 setformats() (translate.convert.ConvertOptionParamethod), 243 setformats() (translate.convert.convert.ConvertOptionParamethod), 243 setformats() (translate.convert.po2moz.MozConvertOptionParamethod), 255	method), 462 arsebetid() (translate.storage.jsonl10n.118NextUnit method), 467 le setid() (translate.storage.jsonl10n.JsonNestedUnit method), 473 le setid() (translate.storage.jsonl10n.JsonUnit method), 476 s- setid() (translate.storage.jsonl10n.WebExtensionJsonUnit reser method), 481 setid() (translate.storage.lisa.LISAunit method), 487 s- setid() (translate.storage.mo.mounit method), 493 setid() (translate.storage.mozilla_lang.LangUnit method), 498 s- setid() (translate.storage.omegat.OmegaTUnit method), 505 setid() (translate.storage.php.LaravelPHPUnit
seterrorleveloptions() (translate.tools.poterminology.TerminologyOptionFormethod), 728 setfilename() (translate.storage.poxliff.PoXliffFirmethod), 579 setfilename() (translate.storage.xliff.xlifffirmethod), 707 setformats() (translate.convert.ArchiveConvertOptionParamethod), 240 setformats() (translate.convert.ConvertOptionParamethod), 243 setformats() (translate.convert.convert.ConvertOptionParamethod), 243 setformats() (translate.convert.po2moz.MozConvertOptionParamethod), 255 setformats() (translate.storage.poxliff.PoXliffFirmethod), 255 setformats() (translate.storage.poxliff.PoXliffFirmethod), 255	method), 462 arsebetid() (translate.storage.jsonl10n.118NextUnit method), 467 le setid() (translate.storage.jsonl10n.JsonNestedUnit method), 473 le setid() (translate.storage.jsonl10n.JsonUnit method), 476 s- setid() (translate.storage.jsonl10n.WebExtensionJsonUnit reser method), 481 setid() (translate.storage.lisa.LISAunit method), 487 s- setid() (translate.storage.mo.mounit method), 493 setid() (translate.storage.mozilla_lang.LangUnit method), 498 s- setid() (translate.storage.omegat.OmegaTUnit method), 505 setid() (translate.storage.php.LaravelPHPUnit method), 562
seterrorleveloptions() (translate.tools.poterminology.TerminologyOptionFormethod), 728 setfilename() (translate.storage.poxliff.PoXliffFirmethod), 579 setfilename() (translate.storage.xliff.xlifffirmethod), 707 setformats() (translate.convert.ArchiveConvertOptionParamethod), 240 setformats() (translate.convert.ConvertOptionParamethod), 243 setformats() (translate.convert.convert.ConvertOptionParamethod), 243 setformats() (translate.convert.po2moz.MozConvertOptionParamethod), 255	method), 462 arsebetid() (translate.storage.jsonl10n.118NextUnit method), 467 le setid() (translate.storage.jsonl10n.JsonNestedUnit method), 473 le setid() (translate.storage.jsonl10n.JsonUnit method), 476 s- setid() (translate.storage.jsonl10n.WebExtensionJsonUnit method), 481 setid() (translate.storage.lisa.LISAunit method), 487 s- setid() (translate.storage.mo.mounit method), 493 setid() (translate.storage.mozilla_lang.LangUnit method), 498 s- setid() (translate.storage.omegat.OmegaTUnit method), 505 setid() (translate.storage.php.LaravelPHPUnit s- method), 562 setid() (translate.storage.php.phpunit method), 567
seterrorleveloptions() (translate.tools.poterminology.TerminologyOptionFormethod), 728 setfilename() (translate.storage.poxliff.PoXliffFirmethod), 579 setfilename() (translate.storage.xliff.xlifffirmethod), 707 setformats() (translate.convert.ArchiveConvertOptionParamethod), 240 setformats() (translate.convert.ConvertOptionParamethod), 243 setformats() (translate.convert.convert.ConvertOptionParamethod), 243 setformats() (translate.convert.po2moz.MozConvertOptionParamethod), 255 setformats() (translate.storage.poxliff.PoXliffFirmethod), 255 setformats() (translate.storage.poxliff.PoXliffFirmethod), 255	method), 462 arsebetid() (translate.storage.jsonl10n.118NextUnit method), 467 le setid() (translate.storage.jsonl10n.JsonNestedUnit method), 473 le setid() (translate.storage.jsonl10n.JsonUnit method), 476 s- setid() (translate.storage.jsonl10n.WebExtensionJsonUnit reser method), 481 setid() (translate.storage.lisa.LISAunit method), 487 s- setid() (translate.storage.mo.mounit method), 493 setid() (translate.storage.mozilla_lang.LangUnit method), 498 s- setid() (translate.storage.omegat.OmegaTUnit method), 505 setid() (translate.storage.php.LaravelPHPUnit method), 562
seterrorleveloptions() (translate.tools.poterminology.TerminologyOptionFormethod), 728 setfilename() (translate.storage.poxliff.PoXliffFirmethod), 579 setfilename() (translate.storage.xliff.xlifffirmethod), 707 setformats() (translate.convert.ArchiveConvertOptionParmethod), 240 setformats() (translate.convert.ConvertOptionParser method), 243 setformats() (translate.convert.ConvertOptionParser method), 243 setformats() (translate.convert.po2moz.MozConvertOptionParser method), 255 setformats() (translate.convert.po2tmx.TmxOptionParser	method), 462 arsebetid() (translate.storage.jsonl10n.I18NextUnit method), 467 le setid() (translate.storage.jsonl10n.JsonNestedUnit method), 473 le setid() (translate.storage.jsonl10n.JsonUnit method), 476 s- setid() (translate.storage.jsonl10n.WebExtensionJsonUnit reser method), 481 setid() (translate.storage.lisa.LISAunit method), 487 s- setid() (translate.storage.mo.mounit method), 493 setid() (translate.storage.mozilla_lang.LangUnit method), 498 s- setid() (translate.storage.omegat.OmegaTUnit method), 505 setid() (translate.storage.php.LaravelPHPUnit method), 562 setid() (translate.storage.php.phpunit method), 567 setid() (translate.storage.pocommon.pounit method),
seterrorleveloptions() (translate.tools.poterminology.TerminologyOptionFormethod), 728 setfilename() (translate.storage.poxliff.PoXliffFirmethod), 579 setfilename() (translate.storage.xliff.xlifffirmethod), 707 setformats() (translate.convert.ArchiveConvertOptionParmethod), 240 setformats() (translate.convert.ConvertOptionParser method), 243 setformats() (translate.convert.convert.ConvertOptionParser method), 255 setformats() (translate.convert.po2moz.MozConvertOptionParser method), 255 setformats() (translate.convert.po2tmx.TmxOptionParser method), 261	method), 462 arsebetid() (translate.storage.jsonl10n.I18NextUnit method), 467 le setid() (translate.storage.jsonl10n.JsonNestedUnit method), 473 le setid() (translate.storage.jsonl10n.JsonUnit method), 476 s- setid() (translate.storage.jsonl10n.WebExtensionJsonUnit reser method), 481 setid() (translate.storage.lisa.LISAunit method), 487 s- setid() (translate.storage.mo.mounit method), 493 setid() (translate.storage.mozilla_lang.LangUnit method), 498 s- setid() (translate.storage.omegat.OmegaTUnit method), 505 setid() (translate.storage.php.LaravelPHPUnit method), 562 setid() (translate.storage.php.phpunit method), 567 setid() (translate.storage.pocommon.pounit method),
seterrorleveloptions() (translate.tools.poterminology.TerminologyOptionEmethod), 728 setfilename() (translate.storage.poxliff.PoXliffFimethod), 579 setfilename() (translate.storage.xliff.xlifffimethod), 707 setformats() (translate.convert.ArchiveConvertOptionParamethod), 240 setformats() (translate.convert.ConvertOptionParsermethod), 243 setformats() (translate.convert.ConvertOptionParsermethod), 255 setformats() (translate.convert.po2tmoz.MozConvertOptionParsermethod), 255 setformats() (translate.convert.po2tmx.TmxOptionParsermethod), 261 setformats() (translate.storage.poxliff.PoXliffFimethod), 261 setformats() (translate.storage.poxliff.PoXliffFimethod), 261 setformats() (translate.storage.poxliff.PoXliffFimethod), 261	method), 462 arsebetid() (translate.storage.jsonl10n.I18NextUnit method), 467 le setid() (translate.storage.jsonl10n.JsonNestedUnit method), 473 le setid() (translate.storage.jsonl10n.JsonUnit method), 476 s- setid() (translate.storage.jsonl10n.WebExtensionJsonUnit method), 481 setid() (translate.storage.lisa.LISAunit method), 487 s- setid() (translate.storage.mo.mounit method), 493 setid() (translate.storage.mozilla_lang.LangUnit method), 498 s- setid() (translate.storage.omegat.OmegaTUnit method), 505 setid() (translate.storage.php.LaravelPHPUnit s- method), 562 setid() (translate.storage.php.phpunit method), 567 setid() (translate.storage.pocommon.pounit method), 567 setid() (translate.storage.pocommon.pounit method), 567
seterrorleveloptions() (translate.tools.poterminology.TerminologyOptionFormethod), 728 setfilename() (translate.storage.poxliff.PoXliffFirmethod), 579 setfilename() (translate.storage.xliff.xlifffirmethod), 707 setformats() (translate.convert.ArchiveConvertOptionParamethod), 240 setformats() (translate.convert.ConvertOptionParseropthod), 243 setformats() (translate.convert.ConvertOptionParseropthod), 255 setformats() (translate.convert.po2moz.MozConvertOptionParseropthod), 261 setformats() (translate.convert.po2tmx.TmxOptionParseropthod), 261 setformats() (translate.convert.po2wordfast.WfOptionParseropthod), 265 setformats() (translate.convert.po2wordfast.WfOptionParseropthod), 265	method), 462 arsebetid() (translate.storage.jsonl10n.118NextUnit method), 467 le setid() (translate.storage.jsonl10n.JsonNestedUnit method), 473 le setid() (translate.storage.jsonl10n.JsonUnit method), 476 s- setid() (translate.storage.jsonl10n.WebExtensionJsonUnit method), 481 setid() (translate.storage.lisa.LISAunit method), 487 s- setid() (translate.storage.mo.mounit method), 493 setid() (translate.storage.mozilla_lang.LangUnit method), 498 s- setid() (translate.storage.omegat.OmegaTUnit method), 505 setid() (translate.storage.php.LaravelPHPUnit s- method), 562 setid() (translate.storage.php.phpunit method), 567 setid() (translate.storage.pocommon.pounit method), 573 setid() (translate.storage.poxliff.PoXliffUnit method), 583 s- setid() (translate.storage.properties.proppluralunit
seterrorleveloptions() (translate.tools.poterminology.TerminologyOptionEmethod), 728 setfilename() (translate.storage.poxliff.PoXliffEmethod), 579 setfilename() (translate.storage.xliff.xlifffilemethod), 707 setformats() (translate.convert.ArchiveConvertOptionParmethod), 240 setformats() (translate.convert.Convert.ConvertOptionParsermethod), 243 setformats() (translate.convert.convert.ConvertOptionParsermethod), 255 setformats() (translate.convert.po2moz.MozConvertOptionParsermethod), 261 setformats() (translate.convert.po2tmx.TmxOptionParsermethod), 261 setformats() (translate.convert.po2wordfast.WfOptionParsermethod), 265	method), 462 arsebetid() (translate.storage.jsonl10n.118NextUnit method), 467 le setid() (translate.storage.jsonl10n.JsonNestedUnit method), 473 le setid() (translate.storage.jsonl10n.JsonUnit method), 476 s- setid() (translate.storage.jsonl10n.WebExtensionJsonUnit method), 481 setid() (translate.storage.lisa.LISAunit method), 487 s- setid() (translate.storage.mo.mounit method), 493 setid() (translate.storage.mozilla_lang.LangUnit method), 498 s- setid() (translate.storage.omegat.OmegaTUnit method), 505 setid() (translate.storage.php.LaravelPHPUnit s- method), 562 setid() (translate.storage.php.phpunit method), 567 setid() (translate.storage.pocommon.pounit method), 573 setid() (translate.storage.poxliff.PoXliffUnit method), 583 s- setid() (translate.storage.properties.proppluralunit
seterrorleveloptions() (translate.tools.poterminology.TerminologyOptionFormethod), 728 setfilename() (translate.storage.poxliff.PoXliffFirmethod), 579 setfilename() (translate.storage.xliff.xlifffirmethod), 707 setformats() (translate.convert.ArchiveConvertOptionParamethod), 240 setformats() (translate.convert.ConvertOptionParseropthod), 243 setformats() (translate.convert.ConvertOptionParseropthod), 255 setformats() (translate.convert.po2moz.MozConvertOptionParseropthod), 261 setformats() (translate.convert.po2tmx.TmxOptionParseropthod), 261 setformats() (translate.convert.po2wordfast.WfOptionParseropthod), 265 setformats() (translate.convert.po2wordfast.WfOptionParseropthod), 265	method), 462 arsebetid() (translate.storage.jsonl10n.118NextUnit method), 467 le setid() (translate.storage.jsonl10n.JsonNestedUnit method), 473 le setid() (translate.storage.jsonl10n.JsonUnit method), 476 s- setid() (translate.storage.jsonl10n.WebExtensionJsonUnit method), 481 setid() (translate.storage.lisa.LISAunit method), 487 s- setid() (translate.storage.mo.mounit method), 493 setid() (translate.storage.mozilla_lang.LangUnit method), 498 s- setid() (translate.storage.omegat.OmegaTUnit method), 505 setid() (translate.storage.php.LaravelPHPUnit s- method), 562 setid() (translate.storage.php.phpunit method), 567 setid() (translate.storage.pocommon.pounit method), 573 setid() (translate.storage.poxliff.PoXliffUnit method), 583 s- setid() (translate.storage.properties.proppluralunit
seterrorleveloptions() (translate.tools.poterminology.TerminologyOptionEmethod), 728 setfilename() (translate.storage.poxliff.PoXliffEmethod), 579 setfilename() (translate.storage.xliff.xlifffilemethod), 707 setformats() (translate.convert.ArchiveConvertOptionPamethod), 240 setformats() (translate.convert.Convert.ConvertOptionParsermethod), 243 setformats() (translate.convert.ConvertOptionParsermethod), 255 setformats() (translate.convert.ConvertOptionParsermethod), 261 setformats() (translate.convert.po2tmx.TmxOptionParsermethod), 265 setformats() (translate.convert.po2wordfast.WfOptionParsermethod), 265 setformats() (translate.convert.po2wordfast.WfOptionParsermethod), 265 setformats() (translate.convert.po2wordfast.WfOptionParsermethod), 265	method), 462 arsebetid() (translate.storage.jsonl10n.I18NextUnit method), 467 le setid() (translate.storage.jsonl10n.JsonNestedUnit method), 473 le setid() (translate.storage.jsonl10n.JsonUnit method), 476 s- setid() (translate.storage.jsonl10n.WebExtensionJsonUnit method), 481 setid() (translate.storage.lisa.LISAunit method), 487 s- setid() (translate.storage.mo.mounit method), 493 setid() (translate.storage.mozilla_lang.LangUnit method), 498 s- setid() (translate.storage.omegat.OmegaTUnit method), 505 setid() (translate.storage.php.LaravelPHPUnit method), 562 setid() (translate.storage.php.phpunit method), 567 setid() (translate.storage.pocommon.pounit method), 583 s- setid() (translate.storage.poxliff.PoXliffUnit method), 583 s- setid() (translate.storage.properties.proppluralunit method), 609 setid() (translate.storage.properties.proppunit

method), 621	late.search.match.terminologymatcher
setid() (translate.storage.pypo.pounit method), 627	method), 402
setid() (translate.storage.qm.qmunit method), 633	setparts() (translate.storage.oo.ooline method), 506
setid() (translate.storage.qph.QphUnit method), 639	setpotoption() (trans-
setid() (translate.storage.rc.rcunit method), 644	late.convert.convert.ArchiveConvertOptionParser
setid() (translate.storage.subtitles.SubtitleUnit	method), 240
method), 658	setpotoption() (trans-
setid() (translate.storage.tbx.tbxunit method), 664	late.convert.convert.ConvertOptionParser
setid() (translate.storage.tiki.TikiUnit method), 669	method), 243
setid() (translate.storage.tmx.tmxunit method), 674	setpotoption() (trans-
setid() (translate.storage.trados.TradosUnit method), 678	late.convert.po2moz.MozConvertOptionParser method), 255
setid() (translate.storage.ts2.tsunit method), 686	setpotoption() (trans-
setid() (translate.storage.txt.TxtUnit method), 691	late.convert.po2tmx.TmxOptionParser
setid() (translate.storage.utx.UtxUnit method), 697	method), 261
setid() (translate.storage.wordfast.WordfastUnit	setpotoption() (trans-
method), 703	late.convert.po2wordfast.WfOptionParser
setid() (translate.storage.xliff.xliffunit method), 710	method), 266
setmanpageoption() (trans-	setprogressoptions() (trans-
late.convert.convert.ArchiveConvertOptionParse method), 240	r late.convert.convert.ArchiveConvertOptionParser method), 240
setmanpageoption() (trans-	setprogressoptions() (trans-
late.convert.convert.ConvertOptionParser	late.convert.convert.ConvertOptionParser
method), 243	method), 244
setmanpageoption() (trans-	setprogressoptions() (trans-
late.convert.po2moz.MozConvertOptionParser method), 255	late.convert.po2moz.MozConvertOptionParser method), 255
setmanpageoption() (trans-	setprogressoptions() (trans-
setmanpageoption() (trans- late.convert.po2tmx.TmxOptionParser	setprogressoptions() (trans- late.convert.po2tmx.TmxOptionParser
late. convert. po 2 tmx. Tmx Option Parser	late.convert.po2tmx.TmxOptionParser
late.convert.po2tmx.TmxOptionParser method), 261	late.convert.po2tmx.TmxOptionParser method), 261
$\begin{array}{c} \textit{late.convert.po2tmx.TmxOptionParser} \\ \textit{method}), 261 \\ \text{setmanpageoption()} & \textit{(trans-} \end{array}$	late.convert.po2tmx.TmxOptionParser method), 261 setprogressoptions() (trans- late.convert.po2wordfast.WfOptionParser method), 266
$late.convert.po2tmx.TmxOptionParser\\method), 261\\setmanpageoption() (translate.convert.po2wordfast.WfOptionParser\\method), 266\\setmanpageoption() (translate.convert.po2wordfast.wfOptionParser)$	late.convert.po2tmx.TmxOptionParser method), 261 setprogressoptions() (trans- late.convert.po2wordfast.WfOptionParser method), 266 setprogressoptions() (trans-
<pre>late.convert.po2tmx.TmxOptionParser method), 261 setmanpageoption() (trans- late.convert.po2wordfast.WfOptionParser method), 266</pre>	late.convert.po2tmx.TmxOptionParser method), 261 setprogressoptions() (trans- late.convert.po2wordfast.WfOptionParser method), 266
<pre>late.convert.po2tmx.TmxOptionParser method), 261 setmanpageoption() (trans- late.convert.po2wordfast.WfOptionParser method), 266 setmanpageoption() (trans- late.filters.pofilter.FilterOptionParser method),</pre>	late.convert.po2tmx.TmxOptionParser method), 261 setprogressoptions() (trans- late.convert.po2wordfast.WfOptionParser method), 266 setprogressoptions() (trans- late.filters.pofilter.FilterOptionParser method),
<pre>late.convert.po2tmx.TmxOptionParser method), 261 setmanpageoption() (trans- late.convert.po2wordfast.WfOptionParser method), 266 setmanpageoption() (trans- late.filters.pofilter.FilterOptionParser method), 355</pre>	late.convert.po2tmx.TmxOptionParser method), 261 setprogressoptions() (trans- late.convert.po2wordfast.WfOptionParser method), 266 setprogressoptions() (trans- late.filters.pofilter.FilterOptionParser method), 355
late.convert.po2tmx.TmxOptionParser method), 261 setmanpageoption() (trans- late.convert.po2wordfast.WfOptionParser method), 266 setmanpageoption() (trans- late.filters.pofilter.FilterOptionParser method), 355 setmanpageoption() (trans- late.misc.optrecurse.RecursiveOptionParser	late.convert.po2tmx.TmxOptionParser method), 261 setprogressoptions() (trans- late.convert.po2wordfast.WfOptionParser method), 266 setprogressoptions() (trans- late.filters.pofilter.FilterOptionParser method), 355 setprogressoptions() (trans- late.misc.optrecurse.RecursiveOptionParser
late.convert.po2tmx.TmxOptionParser method), 261 setmanpageoption() (trans- late.convert.po2wordfast.WfOptionParser method), 266 setmanpageoption() (trans- late.filters.pofilter.FilterOptionParser method), 355 setmanpageoption() (trans- late.misc.optrecurse.RecursiveOptionParser method), 396	late.convert.po2tmx.TmxOptionParser method), 261 setprogressoptions() (trans- late.convert.po2wordfast.WfOptionParser method), 266 setprogressoptions() (trans- late.filters.pofilter.FilterOptionParser method), 355 setprogressoptions() (trans- late.misc.optrecurse.RecursiveOptionParser method), 396
$late.convert.po2tmx.TmxOptionParser\\method), 261\\setmanpageoption() (translate.convert.po2wordfast.WfOptionParser\\method), 266\\setmanpageoption() (translate.filters.pofilter.FilterOptionParser\\method), 355\\setmanpageoption() (translate.misc.optrecurse.RecursiveOptionParser\\method), 396\\setmanpageoption() (translate.tools.poconflicts.ConflictOptionParser}$	late.convert.po2tmx.TmxOptionParser method), 261 setprogressoptions() (trans- late.convert.po2wordfast.WfOptionParser method), 266 setprogressoptions() (trans- late.filters.pofilter.FilterOptionParser method),
$late.convert.po2tmx.TmxOptionParser\\method), 261\\setmanpageoption() (trans-late.convert.po2wordfast.WfOptionParser\\method), 266\\setmanpageoption() (trans-late.filters.pofilter.FilterOptionParser\\method), 355\\setmanpageoption() (trans-late.misc.optrecurse.RecursiveOptionParser\\method), 396\\setmanpageoption() (trans-late.tools.poconflicts.ConflictOptionParser\\method), 718\\$	late.convert.po2tmx.TmxOptionParser method), 261 setprogressoptions() (trans- late.convert.po2wordfast.WfOptionParser method), 266 setprogressoptions() (trans- late.filters.pofilter.FilterOptionParser method), 355 setprogressoptions() (trans- late.misc.optrecurse.RecursiveOptionParser method), 396 setprogressoptions() (trans- late.tools.poconflicts.ConflictOptionParser method), 718
late.convert.po2tmx.TmxOptionParser method), 261 setmanpageoption() (trans- late.convert.po2wordfast.WfOptionParser method), 266 setmanpageoption() (trans- late.filters.pofilter.FilterOptionParser method), 355 setmanpageoption() (trans- late.misc.optrecurse.RecursiveOptionParser method), 396 setmanpageoption() (trans- late.tools.poconflicts.ConflictOptionParser method), 718 setmanpageoption() (trans- late.tools.pogrep.GrepOptionParser method), 721	late.convert.po2tmx.TmxOptionParser method), 261 setprogressoptions() (trans- late.convert.po2wordfast.WfOptionParser method), 266 setprogressoptions() (trans- late.filters.pofilter.FilterOptionParser method), 355 setprogressoptions() (trans- late.misc.optrecurse.RecursiveOptionParser method), 396 setprogressoptions() (trans- late.tools.poconflicts.ConflictOptionParser method), 718 setprogressoptions() (trans- late.tools.pogrep.GrepOptionParser method),
late.convert.po2tmx.TmxOptionParser method), 261 setmanpageoption() (trans- late.convert.po2wordfast.WfOptionParser method), 266 setmanpageoption() (trans- late.filters.pofilter.FilterOptionParser method), 355 setmanpageoption() (trans- late.misc.optrecurse.RecursiveOptionParser method), 396 setmanpageoption() (trans- late.tools.poconflicts.ConflictOptionParser method), 718 setmanpageoption() (trans- late.tools.pogrep.GrepOptionParser method), 721 setmanpageoption() (trans- late.tools.pogreption() (late.convert.po2tmx.TmxOptionParser method), 261 setprogressoptions() (trans- late.convert.po2wordfast.WfOptionParser method), 266 setprogressoptions() (trans- late.filters.pofilter.FilterOptionParser method), 355 setprogressoptions() (trans- late.misc.optrecurse.RecursiveOptionParser method), 396 setprogressoptions() (trans- late.tools.poconflicts.ConflictOptionParser method), 718 setprogressoptions() (trans- late.tools.pogrep.GrepOptionParser method), 722
late.convert.po2tmx.TmxOptionParser method), 261 setmanpageoption() (trans- late.convert.po2wordfast.WfOptionParser method), 266 setmanpageoption() (trans- late.filters.pofilter.FilterOptionParser method), 355 setmanpageoption() (trans- late.misc.optrecurse.RecursiveOptionParser method), 396 setmanpageoption() (trans- late.tools.poconflicts.ConflictOptionParser method), 718 setmanpageoption() (trans- late.tools.pogrep.GrepOptionParser method), 721 setmanpageoption() (trans- late.tools.porestructure.SplitOptionParser method), 725	late.convert.po2tmx.TmxOptionParser method), 261 setprogressoptions() (trans- late.convert.po2wordfast.WfOptionParser method), 266 setprogressoptions() (trans- late.filters.pofilter.FilterOptionParser method),
late.convert.po2tmx.TmxOptionParser method), 261 setmanpageoption() (trans- late.convert.po2wordfast.WfOptionParser method), 266 setmanpageoption() (trans- late.filters.pofilter.FilterOptionParser method),	late.convert.po2tmx.TmxOptionParser method), 261 setprogressoptions() (trans- late.convert.po2wordfast.WfOptionParser method), 266 setprogressoptions() (trans- late.filters.pofilter.FilterOptionParser method), 355 setprogressoptions() (trans- late.misc.optrecurse.RecursiveOptionParser method), 396 setprogressoptions() (trans- late.tools.poconflicts.ConflictOptionParser method), 718 setprogressoptions() (trans- late.tools.pogrep.GrepOptionParser method), 722 setprogressoptions() (trans- late.tools.porestructure.SplitOptionParser method), 725 setprogressoptions() (trans- late.tools.poterminology.TerminologyOptionParser
late.convert.po2tmx.TmxOptionParser method), 261 setmanpageoption() (trans- late.convert.po2wordfast.WfOptionParser method), 266 setmanpageoption() (trans- late.filters.pofilter.FilterOptionParser method), 355 setmanpageoption() (trans- late.misc.optrecurse.RecursiveOptionParser method), 396 setmanpageoption() (trans- late.tools.poconflicts.ConflictOptionParser method), 718 setmanpageoption() (trans- late.tools.pogrep.GrepOptionParser method), 721 setmanpageoption() (trans- late.tools.porestructure.SplitOptionParser method), 725 setmanpageoption() (trans- late.tools.poterminology.TerminologyOptionParser method), 729	late.convert.po2tmx.TmxOptionParser method), 261 setprogressoptions() (trans- late.convert.po2wordfast.WfOptionParser method), 266 setprogressoptions() (trans- late.filters.pofilter.FilterOptionParser method), 355 setprogressoptions() (trans- late.misc.optrecurse.RecursiveOptionParser method), 396 setprogressoptions() (trans- late.tools.poconflicts.ConflictOptionParser method), 718 setprogressoptions() (trans- late.tools.pogrep.GrepOptionParser method), 722 setprogressoptions() (trans- late.tools.porestructure.SplitOptionParser method), 725 setprogressoptions() (trans- late.tools.poterminology.TerminologyOptionParser method), 729
late.convert.po2tmx.TmxOptionParser method), 261 setmanpageoption() (trans- late.convert.po2wordfast.WfOptionParser method), 266 setmanpageoption() (trans- late.filters.pofilter.FilterOptionParser method), 355 setmanpageoption() (trans- late.misc.optrecurse.RecursiveOptionParser method), 396 setmanpageoption() (trans- late.tools.poconflicts.ConflictOptionParser method), 718 setmanpageoption() (trans- late.tools.pogrep.GrepOptionParser method), 721 setmanpageoption() (trans- late.tools.porestructure.SplitOptionParser method), 725 setmanpageoption() (trans- late.tools.poterminology.TerminologyOptionParser method), 729 setparameters() (translate.search.match.matcher	late.convert.po2tmx.TmxOptionParser method), 261 setprogressoptions() (trans- late.convert.po2wordfast.WfOptionParser method), 266 setprogressoptions() (trans- late.filters.pofilter.FilterOptionParser method),
late.convert.po2tmx.TmxOptionParser method), 261 setmanpageoption() (trans- late.convert.po2wordfast.WfOptionParser method), 266 setmanpageoption() (trans- late.filters.pofilter.FilterOptionParser method), 355 setmanpageoption() (trans- late.misc.optrecurse.RecursiveOptionParser method), 396 setmanpageoption() (trans- late.tools.poconflicts.ConflictOptionParser method), 718 setmanpageoption() (trans- late.tools.pogrep.GrepOptionParser method), 721 setmanpageoption() (trans- late.tools.porestructure.SplitOptionParser method), 725 setmanpageoption() (trans- late.tools.poterminology.TerminologyOptionParser method), 729	late.convert.po2tmx.TmxOptionParser method), 261 setprogressoptions() (trans- late.convert.po2wordfast.WfOptionParser method), 266 setprogressoptions() (trans- late.filters.pofilter.FilterOptionParser method),

late.storage.base.TranslationStore method),	late.storage.pocommon.pofile method), 570
410	setprojectstyle() (trans-
setprojectstyle() (trans-	late.storage.poheader.poheader method).
late.storage.catkeys.CatkeysFile method),	575
417	setprojectstyle() (trans-
setprojectstyle() (trans-	late.storage.poxliff.PoXliffFile method), 579
late.storage.csvl10n.csvfile method), 422 setprojectstyle() (translate.storage.dtd.dtdfile	setprojectstyle() (trans- late.storage.properties.gwtfile method), 597
method), 429	setprojectstyle() (trans-
setprojectstyle() (translate.storage.html.htmlfile method), 437	late.storage.properties.javafile method).
setprojectstyle() (trans-	setprojectstyle() (trans-
late.storage.html.POHTMLParser method), 435	late.storage.properties.javautf16file method). 601
$\verb setprojectstyle() \textit{(translate.storage.ical.icalfile)} $	setprojectstyle() (trans-
method), 443	late.storage.properties.javautf8file method).
setprojectstyle() (translate.storage.ini.inifile	603
method), 448	setprojectstyle() (trans-
setprojectstyle() (trans-	late.storage.properties.joomlafile method). 604
late.storage.jsonl10n.ARBJsonFile method), 453	setprojectstyle() (trans-
setprojectstyle() (trans-	late.storage.properties.propfile method).
late.storage.jsonl10n.GoI18NJsonFile	606
method), 458	setprojectstyle() (trans-
setprojectstyle() (trans-	late.storage.properties.stringsfile method).
late.storage.jsonl10n.I18NextFile method),	614
463	setprojectstyle() (trans-
setprojectstyle() (trans-	late.storage.properties.stringsutf8file method).
late.storage.jsonl10n.JsonFile method), 468	616
	setprojectstyle() (trans-
late.storage.jsonl10n.JsonNestedFile method), 470	late.storage.properties.xwikifile method). 618
setprojectstyle() (trans-	setprojectstyle() (trans-
late.storage.jsonl10n.WebExtensionJsonFile method), 478	late.storage.properties.XWikiFullPage method), 593
setprojectstyle() (translate.storage.lisa.LISAfile	setprojectstyle() (trans-
method), 483	late.storage.properties.XWikiPageProperties
setprojectstyle() (translate.storage.mo.mofile method), 490	method), 595
setprojectstyle() (trans-	setprojectstyle() (translate.storage.pypo.pofile method), 624
late.storage.mozilla_lang.LangStore method),	setprojectstyle() (translate.storage.qm.qmfile
495	method), 630
setprojectstyle() (trans-	setprojectstyle() (translate.storage.qph.QphFile
late.storage.omegat.OmegaTFile method),	method), 635
500	setprojectstyle() (translate.storage.rc.rcfile
setprojectstyle() (trans-	method), 641
late.storage.omegat.OmegaTFileTab method), 502	setprojectstyle() (trans- late.storage.subtitles.AdvSubStationAlphaFile
setprojectstyle() (trans-	method), 648
late.storage.php.LaravelPHPFile method),	setprojectstyle() (trans-
559 (translate storage plu plufile	late.storage.subtitles.MicroDVDFile method). 650
setprojectstyle() (translate.storage.php.phpfile method), 564	setprojectstyle() (trans-
setprojectstyle() (trans-	late.storage.subtitles.SubRipFile (mathod).
	1

late.storage.subitiles.SubStationAlphaFile method), 653 setprojectstyle() (translate.storage.tbx.tbxfile method), 655 setprojectstyle() (translate.storage.tbx.tbxfile method), 660 setprojectstyle() (translate.storage.tbx.tbxfile method), 660 setprojectstyle() (translate.storage.tbx.tbxfile method), 660 setprojectstyle() (translate.storage.tbx.tbxfile method), 660 setprojectstyle() (translate.storage.tmx.tmxfile method), 680 setprojectstyle() (translate.storage.tmx.tmxfile method), 680 setprojectstyle() (translate.storage.txt.txtFile method), 682 setprojectstyle() (translate.storage.txt.txtFile method), 683 setprojectstyle() (translate.storage.txt.txtFile method), 693 setprojectstyle() (translate.storage.txt.txtFile method), 495 setsourcel.anguage() (translate.storage.txt.txtFile method), 495 setsourcel.anguage() (translate.storage.txt.txtFile late.storage.pmpeato.tyle() (translate.storage.tyle() (translate.storage.tyle	652	setsourcelanguage() (trans-
setprojectstyle () (translate.storage.tbx.tbxfile method), 660 setprojectstyle () (translate.storage.tiki.TikiStore method), 660 setprojectstyle () (translate.storage.tiki.TikiStore method), 660 setprojectstyle () (translate.storage.tiki.TikiStore method), 660 setprojectstyle () (translate.storage.tbx.tbxfile method), 660 setprojectstyle () (translate.storage.tbx.tbxfile method), 680 setprojectstyle () (translate.storage.tsz.txfile method), 688 setprojectstyle () (translate.storage.tsz.txfile method), 688 setprojectstyle () (translate.storage.txt.Txfile method), 693 setprojectstyle () (translate.storage.txt.Txfile method), 693 setprojectstyle () (translate.storage.txt.txfile method), 693 setprojectstyle () (translate.storage.txt.fxfile method), 700 setprojectstyle () (translate.storage.txt.fxfile method), 700 setprojectstyle () (translate.storage.txt.fxfile method), 405 setsourcelanguage () (translate.storage.mozillifx.liffile method), 405 setsourcelanguage () (translate.storage.php.phpfile method), 570 setsourcelanguage () (translate.storage.php.phpfile	<pre>setprojectstyle() (trans- late.storage.subtitles.SubStationAlphaFile</pre>	late.storage.jsonl10n.I18NextFile method), 463
late.storage.subitiles.SubitileFile method), 660 setprojectstyle() (translate.storage.tix.tbxfile method), 660 setprojectstyle() (translate.storage.tix.tTikiStore method), 661 setprojectstyle() (translate.storage.tix.tmxfile method), 662 setprojectstyle() (translate.storage.tix.tixlistore method), 663 setprojectstyle() (translate.storage.txt.Tixlistore method), 682 setprojectstyle() (translate.storage.txt.Tixlistore method), 683 setprojectstyle() (translate.storage.txt.Tixlistore method), 700 setprojectstyle() (translate.storage.txt.Tixlistore m	method), 653	
late.storage.json/10n_JsonNestedFile method), 660 setprojectstyle() (translate.storage.tiki.TikiStore method), 660 setprojectstyle() (translate.storage.tiki.TikiStore method), 661 setprojectstyle() (translate.storage.tmx.tmxfile method), 680 setprojectstyle() (translate.storage.txt.TxtFile method), 682 setprojectstyle() (translate.storage.txt.TxtFile method), 682 setprojectstyle() (translate.storage.txt.TxtFile method), 688 setprojectstyle() (translate.storage.txt.TxtFile method), 688 setprojectstyle() (translate.storage.txt.TxtFile method), 693 setprojectstyle() (translate.storage.txt.TxtFile method), 694 setsourcelanguage() (translate.storage.method), 695 setsourcelanguage() (translate.storage.txt.TxtFile method), 695 setsourcelanguage() (translate.storage.txt.TxtFile method), 700 setsourcelanguage() (translate.storage.txt.TxtFile method), 405 setsourcelanguage() (translate.storage.txt.T		late.storage.jsonl10n.JsonFile method), 468
method), 660 setprojectstyle() (translate.storage.tixi.TikiStore method), 660 setprojectstyle() (translate.storage.tixi.mx.file method), 671 setprojectstyle() (translate.storage.tixi.mx.file method), 680 setprojectstyle() (translate.storage.tixi.TikiStore method), 680 setprojectstyle() (translate.storage.tixi.TikiStore method), 680 setprojectstyle() (translate.storage.tixi.TikiStore method), 682 setprojectstyle() (translate.storage.tixi.TikiStore method), 682 setprojectstyle() (translate.storage.tixi.TikiStore method), 683 setprojectstyle() (translate.storage.tixi.TikiStore method), 683 setprojectstyle() (translate.storage.tixi.TikiStore method), 683 setprojectstyle() (translate.storage.tixi.TikiStore method), 683 setprojectstyle() (translate.storage.tixi.TikiStore method), 680 setprojectstyle() (translate.storage.tixi.TikiStore method), 680 setprojectstyle() (translate.storage.tixi.TikiStore method), 495 setprojectstyle() (translate.storage.tixi.TikiStore method), 405 setprojectstyle() (translate.storage.tixi.TikiStore method), 405 setprojectstyle() (translate.storage.tixi.TikiStore method), 405 setprojectstyle() (translate.storage.tixi.TikiStore method), 405 setprojectstyle() (tra		
late.storage_jsonllOn.WebExtensionJsonFile method), 671		***
method), 671 setprojectstyle() (translate.storage.tx2.txfile method), 682 setprojectstyle() (translate.storage.tx1.TxtFile method), 688 setprojectstyle() (translate.storage.tx1.TxtFile method), 688 setprojectstyle() (translate.storage.utx.UtxFile method), 693 setprojectstyle() (translate.storage.utx.UtxFile method), 700 setprojectstyle() (translate.storage.xtiff.xtiffile method), 400 setprojectstyle() (translate.storage.xtiff.xtiffile method), 693 setprojectstyle() (translate.storage.xtiff.xtiffile method), 600 setporiectstyle() (translate.storage.xtiff.xtiffile method), 600 setporiectstyle() (translate.storage.xtiff.xtiffile method), 600 setporiectstyle() (translate.storage.xtiff.xtiffile method), 600 setporiectstyle() (translate.storage.xtiff.xtiffile method), 400 setporiectstyle() (translate.storage.xtiff.xtiffile method), 400 setporiectstyle() (translate.storage.xtiff.xtiffile method), 400 setporiectstyle() (translate.storage.xtiff.xtiffile method), 400 setporiectstyle() (translate.storage.xt.or.xtifile method), 400 setporiectstyle() (translate.storage.xtif	setprojectstyle() (translate.storage.tiki.TikiStore	late. storage. js on 110 n. Web Extension Js on File
setprojectstyle() (translate.storage.txt.TxtFile method), 682 setprojectstyle() (translate.storage.txt.TxtFile method), 688 setprojectstyle() (translate.storage.txt.TxtFile method), 688 setprojectstyle() (translate.storage.txt.TxtFile method), 689 setprojectstyle() (translate.storage.txt.TxtFile method), 689 setprojectstyle() (translate.storage.txt.TxtFile method), 700 setprojectstyle() (translate.storage.txt.TxtFile method), 700 setprojectstyle() (translate.storage.xtiff.xtiffile method), 700 setsourcelanguage() (translate.storage.xtiff.xtiffile method), 700 setsourcelanguage() (translate.storage.txt.TxtFile method), 850 setsourcelanguage() (translate.storage.txt.TxtFile setsourcelanguage() (translate.storage.txt.TxtFile setsourcelanguage() (translate.storage.onegat.OmegatFile method), 500 setsourcelanguage() (translate.storage.txt.TxtFile setsourcelanguage()		
late.storage.trados.TradosTxtTmFile method), 680		
setprojectstyle () (translate.storage.txt.TxtFile method), 682 setprojectstyle () (translate.storage.uxt.UtxFile method), 693 setprojectstyle () (translate.storage.uxt.UtxFile method), 693 setprojectstyle () (translate.storage.worlfast.WordfastTMFile method), 700 setsourcelanguage () (translate.storage.xkiff.xkifffile method), 707 setsourcelanguage () (translate.storage.base.DictStore method), 405 setsourcelanguage () (translate.storage.catkeys.CatkeysFile method), 410 setsourcelanguage () (translate.storage.catkeys.CatkeysFile method), 421 setsourcelanguage () (translate.storage.catkeys.CatkeysFile method), 422 setsourcelanguage () (translate.storage.catkeys.CatkeysFile method), 422 setsourcelanguage () (translate.storage.dintl.htmlfile method), 427 setsourcelanguage () (translate.storage.catkeys.CatkeysFile method), 429 setsourcelanguage () (translate.storage.dintl.htmlfile method), 437 setsourcelanguage () (translate.storage.dintl.htmlfile method), 437 setsourcelanguage () (translate.storage.dintl.htmlfile method), 437 setsourcelanguage () (translate.storage.dintl.htmlfile method), 438 setsourcelanguage () (translate.storage.mini.nifile method), 448 setsourcelanguage () (translate.storage.properties.joomlafile method), 443 setsourcelanguage () (translate.storage.mini.nifile method), 448 setsourcelanguage () (translate.storage.properties.propfile method), 450 setsourcelanguage () (translate.storage.properties.propfile method), 450 setsourcelanguage () (translate.storage.properties.giavautfl6file method), 440 setsourcelanguage () (translate.storage.properties.giavautfl6file met	$late.storage.trados.TradosTxtTmFile \ \ method),$	method), 490
method), 688 setprojectstyle() (translate.storage.utx.UtxFile method), 693 setprojectstyle() (translate.storage.utx.UtxFile setprojectstyle() (translate.storage.wordfast.WordfastTMFile method), 700 setprojectstyle() (translate.storage.xliff.xliffile method), 707 setsourcelanguage() (trans- late.storage.base.DictStore method), 405 setsourcelanguage() (trans- late.storage.base.DictStore method), 405 setsourcelanguage() (trans- late.storage.base.TranslationStore late.storage.base.TranslationStore late.storage.catkeys.CatkeysFile alte.storage.pocommon.pofile method), 570 setsourcelanguage() (trans- late.storage.poxiff.PoXliffFile method), 570 setsourcelanguage() (trans- late.storage.poxiff.PoXliffFile method), 577 setsourcelanguage() (trans- late.storage.properties.gwfile method), 597 setsourcelanguage() (trans- late.storage.properties.javatiflofile method), 429 setsourcelanguage() (trans- late.storage.properties.javatiflofile method), 429 setsourcelanguage() (trans- late.storage.properties.javatiflofile method), 435 setsourcelanguage() (trans- late.storage.properties.javatiflofile method), 448 setsourcelanguage() (trans- late.storage.properties.joomlafile method), 448 setsourcelanguage() (trans- late.storage.properties.propfile method), 449 setsourcelanguage() (trans- late.storage.properties.stringsfile method), 440 setsourcelanguage() (trans- late.	setprojectstyle() (translate.storage.ts2.tsfile	$late. storage. mozilla_lang. Lang Store method),$
setprojectstyle() (translate.storage.utx.UtxFile method), 693 setprojectstyle() (translate.storage.wordfast.WordfastTMFile method), 700 setprojectstyle() (translate.storage.xliff.xlifffile method), 700 setprojectstyle() (translate.storage.xliff.xlifffile method), 707 setprojectstyle() (translate.storage.xliff.xlifffile method), 707 setsourcelanguage() (translate.storage.base.DictStore method), 405 setsourcelanguage() (translate.storage.base.DictStore method), 405 setsourcelanguage() (translate.storage.base.TranslationStore method), 410 setsourcelanguage() (translate.storage.catkeys.CatkeysFile method), 417 setsourcelanguage() (translate.storage.catkeys.CatkeysFile method), 422 setsourcelanguage() (translate.storage.dat.dtdfile method), 429 setsourcelanguage() (translate.storage.min.imfile method), 437 setsourcelanguage() (translate.storage.imi.imfile method), 435 setsourcelanguage() (translate.storage.imi.imfile method), 436 setsourcelanguage() (translate.storage.imi.imfile method), 448 setsourcelanguage() (translate.storage.imi.imfile method), 448 setsourcelanguage() (translate.storage.properties.joomlafile method), 443 setsourcelanguage() (translate.storage.properties.propfile method), 606 setsourcelanguage() (translate.storage.properties.propfile method), 606 setsourcelanguage() (translate.storage.properties.propfile method), 606 setsourcelanguage() (translate.storage.prop		
setprojectstyle() (translate.storage.wordfast.WordfastTMFile method), 700 setprojectstyle() (translate.storage.xliff.xlifffile method), 700 setsourcelanguage() (translate.storage.buse.DictStore method), 405 setsourcelanguage() (translate.storage.buse.DictStore method), 405 setsourcelanguage() (translate.storage.buse.TranslationStore method), 405 setsourcelanguage() (translate.storage.buse.TranslationStore method), 410 setsourcelanguage() (translate.storage.catkeys.CatkeysFile method), 417 setsourcelanguage() (translate.storage.catkeys.CatkeysFile method), 427 setsourcelanguage() (translate.storage.dtd.dtdfile method), 429 setsourcelanguage() (translate.storage.html.htmlfile method), 437 setsourcelanguage() (translate.storage.html.htmlfile method), 437 setsourcelanguage() (translate.storage.html.POHTMLParser method), 435 setsourcelanguage() (translate.storage.ini.inifile method), 448 setsourcelanguage() (translate.storage.ini.inifile method), 448 setsourcelanguage() (translate.storage.ini.inifile method), 448 setsourcelanguage() (translate.storage.ini.inifile method), 448 setsourcelanguage() (translate.storage.ini.inifile method), 453 setsourcelanguage() (translate.storage.ini.inifile method), 448 setsourcelanguage() (translate.storage.ini.inifile method), 453 setsourcelanguage() (translate.storage.properties.joomlafile method), 453 setsourcelanguage() (translate.storage.properties.stringsfile method), 453 setsourcelanguage() (translate.storage.properties.stringsfile method), 453 setsourcelanguage() (transla		
late.storage.wordfast.WordfastTMFile method), 700 setprojectstyle() (translate.storage.xliff.xlifffile method), 707 setsourcelanguage() (translate.storage.base.DictStore method), 405 setsourcelanguage() (translate.storage.base.DictStore method), 405 setsourcelanguage() (translate.storage.base.TranslationStore method), 410 setsourcelanguage() (translate.storage.catkeys.CatkeysFile method), 417 setsourcelanguage() (translate.storage.catkeys.CatkeysFile method), 422 setsourcelanguage() (translate.storage.catkeys.CatkeysFile method), 422 setsourcelanguage() (translate.storage.cyvil/On.csvfile method), 422 setsourcelanguage() (translate.storage.properties.gwtfile method), 579 setsourcelanguage() (translate.storage.properties.javautflofile method), 429 setsourcelanguage() (translate.storage.html.htmlfile method), 437 setsourcelanguage() (translate.storage.html.htmlfile method), 437 setsourcelanguage() (translate.storage.properties.javautflofile method), 435 setsourcelanguage() (translate.storage.properties.javautflofile method), 435 setsourcelanguage() (translate.storage.properties.javautflofile method), 448 setsourcelanguage() (translate.storage.properties.joomlafile method), 448 setsourcelanguage() (translate.storage.properties.propfile method), 606 setsourcelanguage() (translate.storage.properties.propfile method), 606 setsourcelanguage() (translate.storage.properties.stringsfile method), 606 setsourcelanguage() (translate.storage.properties.stringsfile method), 606 setsourcelanguage() (translate.storage.properties.stringsfile method), 606		setsourcelanguage() (trans-
setsourcelanguage () (translate.storage.xliff.xlifffile method), 707 setsourcelanguage () (translate.storage.base.DictStore method), 405 setsourcelanguage () (translate.storage.base.TranslationStore date.storage.base.TranslationStore date.storage.base.TranslationStore date.storage.catkeys.CatkeysFile date.storage.catkeys.CatkeysFile date.storage.catkeys.CatkeysFile method), 417 setsourcelanguage () (translate.storage.catkeys.CatkeysFile method), 422 setsourcelanguage () (translate.storage.csvl10n.csvfile method), 422 setsourcelanguage () (translate.storage.did.didfile method), 429 setsourcelanguage () (translate.storage.timl.himfile method), 437 setsourcelanguage () (translate.storage.timl.himfile method), 437 setsourcelanguage () (translate.storage.timl.nimfile method), 437 setsourcelanguage () (translate.storage.properties.javautf16file method), 435 setsourcelanguage () (translate.storage.properties.javautf8file method), 435 setsourcelanguage () (translate.storage.properties.javautf8file method), 438 setsourcelanguage () (translate.storage.properties.javautf8file method), 448 setsourcelanguage () (translate.storage.properties.javautf8file method), 448 setsourcelanguage () (translate.storage.properties.javautf8file method), 448 setsourcelanguage () (translate.storage.properties.javautf8file method), 453 setsourcelanguage () (translate.storage.properties.javautf8file method), 453 setsourcelanguage () (translate.storage.properties.javautf8file method), 605 setsourcelanguage () (translate.storage.properties.propfile method), 606 setsourcelanguage () (translate.storage.properties.propfile method), 606 setsourcelanguage () (translate.storage.properties.gropfile method), 606 setsourcelanguage () (translat		
setsprojectstyle() (translate.storage.xliff.xlifffile method), 707 setsourcelanguage() (translate.storage.base.DictStore method), 405 setsourcelanguage() (translate.storage.base.TranslationStore alte.storage.base.TranslationStore alte.storage.base.TranslationStore alte.storage.catkeys.CatkeysFile method), 410 setsourcelanguage() (translate.storage.catkeys.CatkeysFile method), 417 setsourcelanguage() (translate.storage.catkeys.CatkeysFile method), 429 setsourcelanguage() (translate.storage.dtd.dtdfile method), 429 setsourcelanguage() (translate.storage.dtd.dtdfile late.storage.properties.javafile method), 437 setsourcelanguage() (translate.storage.properties.javautf16file method), 435 setsourcelanguage() (translate.storage.properties.javautf8file method), 435 setsourcelanguage() (translate.storage.min.inifile method), 448 setsourcelanguage() (translate.storage.min.inifile method), 448 setsourcelanguage() (translate.storage.min.inifile method), 448 setsourcelanguage() (translate.storage.properties.joomlafile method), 448 setsourcelanguage() (translate.storage.properties.propfile method), 606 setsourcelanguage() (translate.storage.properties.propfile method), 570 setsourcelanguage() (translate.storage.properties.giavautf16file method), 597 setsourcelanguage() (translate.storage.properties.javautf16file method), 601 setsourcelanguage() (translate.storage.properties.javautf8file method), 605 setsourcelanguage() (translate.storage.properties.propfile method), 606 setsourcelanguage() (translate.storage.properties.propfile method), 570 setsourcelanguage() (translate.storage.properties.javautf16file method), 597 setsourcelanguage() (translate.storage.properties.javautf16file method), 601 setsourcelanguage() (translate.storage.properties.propfile method), 601 setsourcelanguage() (translate.storage.properties.giomlafile method), 605 setsourcelanguage() (translate.storage.properties.propfile method), 606 setsourcelanguage() (translate.storage.properties.propfile method), 606 setsourcelanguage() (t		
method), 707 setsourcelanguage() [trans-late.storage.base.DictStore method), 405 setsourcelanguage() [trans-late.storage.base.DictStore method), 405 setsourcelanguage() [trans-late.storage.base.TranslationStore method), 410 setsourcelanguage() [trans-late.storage.catkeys.CatkeysFile method), 417 setsourcelanguage() [trans-late.storage.catkeys.CatkeysFile method), 422 setsourcelanguage() [trans-late.storage.csvl10n.csvfile method), 422 setsourcelanguage() [trans-late.storage.csvl10n.csvfile method), 422 setsourcelanguage() [trans-late.storage.csvl10n.csvfile method), 422 setsourcelanguage() [trans-late.storage.html.htmlfile method), 437 setsourcelanguage() [trans-late.storage.html.htmlfile method), 437 setsourcelanguage() [trans-late.storage.html.POHTMLParser method), 435 setsourcelanguage() [trans-late.storage.properties.javautf16file method), 436 setsourcelanguage() [trans-late.storage.properties.javautf16file method), 437 setsourcelanguage() [trans-late.storage.properties.javautf16file method), 438 setsourcelanguage() [trans-late.storage.properties.javautf16file method), 439 setsourcelanguage() [trans-late.storage.properties.javautf16file method), 430 setsourcelanguage() [trans-late.storage.properties.javautf16file method), 435 setsourcelanguage() [trans-late.storage.properties.javautf16file method), 435 setsourcelanguage() [trans-late.storage.properties.javautf16file method), 603 setsourcelanguag		
late.storage.base.DictStore method), 405late.storage.base.TranslationStore method), 405late.storage.point.storage.pop.phpfile method), 564setsourcelanguage()(translate.storage.pocommon.pofile method), 570410setsourcelanguage()(translate.storage.pocommon.pofile method), 579setsourcelanguage()(translate.storage.pocommon.pofile method), 579late.storage.catkeys.CatkeysFilemethod),setsourcelanguage()(translate.storage.properties.gwtfile method), 597setsourcelanguage()(translate.storage.properties.gwtfile method), 422setsourcelanguage()(translate.storage.properties.javafilemethod),setsourcelanguage()(translate.storage.properties.javautf10filemethod),setsourcelanguage()(translate.storage.properties.javautf10filemethod),setsourcelanguage()(translate.storage.properties.javautf10filemethod),setsourcelanguage()(translate.storage.properties.javautf10filemethod),setsourcelanguage()(translate.storage.properties.javautf10filemethod),setsourcelanguage()(translate.storage.properties.joomlafilemethod),setsourcelanguage()(translate.storage.properties.joomlafilemethod),setsourcelanguage()(translate.storage.properties.propfilemethod),setsourcelanguage()(translate.storage.properties.propfilemethod),setsourcelanguage()(translate.storage.properties.stringsfilemethod),setsourcelanguage()(translate.storage.properties.stringsfilemethod),setsourcelanguage()(translate.storage.properties.stringsfile	method), 707	559
setsourcelanguage() (trans- late.storage.base.TranslationStore method), 410 setsourcelanguage() (trans- late.storage.catkeys.CatkeysFile method), 417 setsourcelanguage() (trans- late.storage.catkeys.CatkeysFile method), 427 setsourcelanguage() (trans- late.storage.catkeys.CatkeysFile method), 429 setsourcelanguage() (translate.storage.dtd.dtdfile method), 429 setsourcelanguage() (translate.storage.dtd.dtdfile method), 437 setsourcelanguage() (translate.storage.html.hmlfile method), 437 setsourcelanguage() (translate.storage.html.POHTMLParser method), 435 setsourcelanguage() (translate.storage.ical.icalfile method), 443 setsourcelanguage() (translate.storage.ini.inifile method), 448 setsourcelanguage() (translate.storage.properties.propfile method), 606 setsourcelanguage() (translate.storage.properties.stringsfile method), 606 setsourcelanguage() (translate.storage.properties.gvatife method), 606 setsourcelanguage() (translate.storage.properties.gvatife method), 606 setsourcelanguage() (translate.storage.properties.gvatife method), 606 setsourcelanguage() (translate.storage.properties.gvatife method), 606 setsour		
late.storage.base.TranslationStore 410 setsourcelanguage () (transsetsourcelanguage () (translate.storage.catkeys.CatkeysFile method), 579 late.storage.catkeys.CatkeysFile method), 579 setsourcelanguage () (translate.storage.catkeys.CatkeysFile method), 417 setsourcelanguage () (translate.storage.cvtl10n.csvfile method), 422 setsourcelanguage () (translate.storage.dtd.dtdfile method), 429 setsourcelanguage () (translate.storage.dtd.dtdfile method), 437 setsourcelanguage () (translate.storage.html.htmlfile method), 437 setsourcelanguage () (translate.storage.html.htmlfile method), 435 setsourcelanguage () (translate.storage.ini.inifile method), 448 setsourcelanguage () (translate.storage.ini.inifile method), 448 setsourcelanguage () (translate.storage.jsonl10n.ARBJsonFile method), 453 setsourcelanguage () (translate.storage.jsonl10n.GoI18NJsonFile method), 464 late.storage.properties.stringsfile method), 465 setsourcelanguage () (translate.storage.properties.stringsfile method), 464 method), 465 setsourcelanguage () (translate.storage.properties.stringsfile method), 465 setsourcelanguage () (translate.storage.properties.stringsfile method), 466 setsourcelanguage () (translate.storage.properties.stringsfile method), 467 setsourcelanguage () (translate.storage.properties.stringsfile method), 468 setsourcelanguage () (translate.storage.properties.stringsfile method), 469 setsourcelanguage () (translate.storage.properties.gwtfile method), 469 setsourcelanguage () (translate.storage.properties.javautf8file method), 469 setsourcelanguage () (translate.storage.properties.javautf8file method), 469 setsourcelanguage () (translate.storage.properties.javautf8file method), 469 setsourcelanguage () (translate.storage.propertie		
410setsourcelanguage()(translate.storage.poxliff.PoXliffFile method), 579late.storage.catkeys.CatkeysFilemethod), 417setsourcelanguage()(translate.storage.properties.gwtfile method), 597setsourcelanguage()(translate.storage.csv110n.csvfile method), 422setsourcelanguage()(translate.storage.properties.javafilemethod), 597setsourcelanguage()(translate.storage.properties.javafilemethod), 429setsourcelanguage()(translate.storage.properties.javautf16filemethod), 437setsourcelanguage()(translate.storage.properties.javautf16filemethod), 435setsourcelanguage()(translate.storage.properties.javautf8filemethod), 603setsourcelanguage()(translate.storage.properties.javautf8filemethod), 603setsourcelanguage()(translate.storage.properties.joomlafilemethod), 603setsourcelanguage()(translate.storage.properties.joomlafilemethod), 605setsourcelanguage()(translate.storage.properties.propfilemethod), 606setsourcelanguage()(translate.storage.properties.propfilemethod), 606setsourcelanguage()(translate.storage.properties.propfilemethod), 606setsourcelanguage()(translate.storage.properties.stringsfilemethod), 606setsourcelanguage()(translate.storage.properties.stringsfilemethod), 606setsourcelanguage()(translate.storage.properties.stringsfilemethod), 606setsourcelanguage()(translate.storage.properties.stringsfilelate.storage.jsonl10n.GoI18NJsonFile(translate.storage.properties.stringsfilemethod), 606		
setsourcelanguage() (translate.storage.catkeys.CatkeysFile method), 579 setsourcelanguage() (translate.storage.csvl10n.csvfile method), 422 setsourcelanguage() (translate.storage.csvl10n.csvfile method), 422 setsourcelanguage() (translate.storage.dtd.dtdfile method), 429 setsourcelanguage() (translate.storage.dtd.dtdfile method), 437 setsourcelanguage() (translate.storage.html.htmlfile method), 437 setsourcelanguage() (translate.storage.html.POHTMLParser method), 435 setsourcelanguage() (translate.storage.html.pOHTMLParser method), 435 setsourcelanguage() (translate.storage.ini.inifile method), 448 setsourcelanguage() (translate.storage.ini.inifile method), 448 setsourcelanguage() (translate.storage.jsonl10n.ARBJsonFile method), 606 setsourcelanguage() (translate.storage.properties.propfile method), 606 setsourcelanguage() (translate.storage.properties.stringsfile method), 606		
late.storage.catkeys.CatkeysFile method), 417 setsourcelanguage() (translate.storage.csvlIOn.csvfile method), 422 setsourcelanguage() (translate.storage.dtd.dtdfile method), 429 setsourcelanguage() (translate.storage.dtd.dtdfile method), 429 setsourcelanguage() (translate.storage.html.htmlfile method), 437 setsourcelanguage() (translate.storage.html.htmlfile method), 437 setsourcelanguage() (translate.storage.html.POHTMLParser method), 435 setsourcelanguage() (translate.storage.ini.inifile method), 443 setsourcelanguage() (translate.storage.ini.inifile method), 448 setsourcelanguage() (translate.storage.ini.inifile method), 448 setsourcelanguage() (translate.storage.ini.inifile method), 448 setsourcelanguage() (translate.storage.jsonl10n.ARBJsonFile method), 606 setsourcelanguage() (translate.storage.jsonl10n.Gol18NJsonFile method), 614		
417		
setsourcelanguage() (trans- late.storage.csvl10n.csvfile method), 422 late.storage.properties.javafile method), setsourcelanguage() (translate.storage.dtd.dtdfile method), 429 setsourcelanguage() (translate.storage.html.htmlfile method), 437 setsourcelanguage() (translate.storage.html.POHTMLParser method), 435 setsourcelanguage() (translate.storage.ical.icalfile method), 443 setsourcelanguage() (translate.storage.ical.icalfile method), 443 setsourcelanguage() (translate.storage.ini.inifile method), 448 setsourcelanguage() (translate.storage.joonl10n.ARBJsonFile method), 453 setsourcelanguage() (translate.storage.joonl10n.GoI18NJsonFile method), 606 setsourcelanguage() (translate.storage.properties.stringsfile method), 606 setsourcelanguage() (translate.storage.properties.stringsfile method), 614	· ·	
setsourcelanguage() (translate.storage.dtd.dtdfile method), 429 setsourcelanguage() (trans- late.storage.html.htmlfile method), 437 setsourcelanguage() (trans- late.storage.html.POHTMLParser method), 435 setsourcelanguage() (trans- late.storage.ical.icalfile method), 443 setsourcelanguage() (trans- late.storage.ini.inifile method), 448 setsourcelanguage() (trans- late.storage.jsonl10n.ARBJsonFile method), d53 setsourcelanguage() (trans- late.storage.jsonl20n.Gol18NJsonFile method), d606 setsourcelanguage() (trans- late.storage.properties.propfile method), d606 setsourcelanguage() (trans- late.storage.properties.propfile method), d606 setsourcelanguage() (trans- late.storage.properties.stringsfile method), d606	setsourcelanguage() (trans-	
method), 429setsourcelanguage()(trans- late.storage.properties.javautf16file(trans- method),setsourcelanguage()(trans- late.storage.html.htmlfile method), 437601(trans- 		late.storage.properties.javafile method),
setsourcelanguage() (translate.storage.html.htmlfile method), 437 601 setsourcelanguage() (translate.storage.html.POHTMLParser method), 435 setsourcelanguage() (translate.storage.ical.icalfile method), 443 setsourcelanguage() (translate.storage.ini.inifile method), 448 setsourcelanguage() (translate.storage.ini.inifile method), 448 setsourcelanguage() (translate.storage.jsonl10n.ARBJsonFile method), 453 setsourcelanguage() (translate.storage.jsonl10n.GoI18NJsonFile method), 4614 method), 614		599
late.storage.html.htmlfile method), 437601setsourcelanguage()(trans- late.storage.html.POHTMLParser method), 435setsourcelanguage() late.storage.properties.javautf8file 603method), (trans- late.storage.properties.joomlafile method),setsourcelanguage()(trans- late.storage.properties.joomlafile setsourcelanguage()method), (trans- late.storage.properties.propfile method),setsourcelanguage()(trans- late.storage.properties.propfile method), 606 setsourcelanguage()method), (trans- late.storage.properties.stringsfile late.storage.properties.stringsfilesetsourcelanguage()(trans- late.storage.properties.stringsfile late.storage.properties.stringsfilemethod), method), late.storage.properties.stringsfile		3 3
setsourcelanguage() (trans- late.storage.html.POHTMLParser method), 435 (one of the setsourcelanguage) (trans- late.storage.ical.icalfile method), 443 (trans- late.storage.ical.icalfile method), 443 (setsourcelanguage) (translate.storage.ini.inifile method), 448 (setsourcelanguage) (translate.storage.ini.inifile setsourcelanguage) (translate.storage.jni.inifile late.storage.jsonl10n.ARBJsonFile method), late.storage.jsonl10n.ARBJsonFile method), late.storage.jsonl10n.GoI18NJsonFile method), late.storage.jsonl10n.GoI18NJsonFile method), late.storage.jsonl10n.GoI18NJsonFile method), late.storage.jsonl10n.GoI18NJsonFile method), late.storage.jsonl10n.GoI18NJsonFile method), late.storage.jsonl10n.GoI18NJsonFile method),		
late.storage.html.POHTMLParsermethod),late.storage.properties.javautf8filemethod),435603(trans-setsourcelanguage()(trans-setsourcelanguage()(trans-late.storage.ical.icalfile method), 443late.storage.properties.joomlafilemethod),setsourcelanguage()(trans-605method), 448setsourcelanguage()(trans-setsourcelanguage()(trans-late.storage.properties.propfilemethod),late.storage.jsonl10n.ARBJsonFilemethod),606setsourcelanguage()(trans-late.storage.properties.stringsfilemethod),late.storage.jsonl10n.GoI18NJsonFile614	· ·	
435 603 setsourcelanguage() (trans- late.storage.ical.icalfile method), 443 late.storage.properties.joomlafile method), setsourcelanguage() (translate.storage.ini.inifile method), 448 setsourcelanguage() (translate.storage.properties.propfile method), late.storage.jsonl10n.ARBJsonFile method), late.storage.jsonl10n.ARBJsonFile method), late.storage.jsonl10n.GoI18NJsonFile 614 603 (translate.storage.properties.joomlafile method), (translate.storage.properties.propfile method), late.storage.jsonl10n.ARBJsonFile method), late.storage.jsonl10n.GoI18NJsonFile 614		
late.storage.ical.icalfile method), 443late.storage.properties.joomlafile method),setsourcelanguage() (translate.storage.ini.inifile method), 448setsourcelanguage()(transsetsourcelanguage()setsourcelanguage()(translate.storage.properties.propfile method),method),late.storage.jsonl10n.ARBJsonFile method),606453setsourcelanguage()(transsetsourcelanguage()setsourcelanguage()(translate.storage.properties.stringsfile method),late.storage.jsonl10n.GoI18NJsonFile614		
late.storage.ical.icalfile method), 443late.storage.properties.joomlafile method),setsourcelanguage() (translate.storage.ini.inifile method), 448setsourcelanguage()(transsetsourcelanguage()setsourcelanguage()(translate.storage.properties.propfile method),method),late.storage.jsonl10n.ARBJsonFile method),606(transsetsourcelanguage()(transsetsourcelanguage()(transsetsourcelanguage, properties.stringsfile method),setsourcelanguage()(translate.storage.properties.stringsfile method),late.storage.properties.stringsfile method),	setsourcelanguage() (trans-	setsourcelanguage() (trans-
method), 448setsourcelanguage()(transsetsourcelanguage)(translate.storage.properties.propfilemethod),setsourcelanguage()method),606453setsourcelanguage()(transsetsourcelanguage)(translate.storage.properties.stringsfilemethod),late.storage.jsonl10n.GoI18NJsonFile614		late.storage.properties.joomlafile method),
$\begin{tabular}{lllllllllllllllllllllllllllllllllll$	<pre>setsourcelanguage() (translate.storage.ini.inifile</pre>	605
late.storage.jsonl10n.ARBJsonFilemethod),606453setsourcelanguage()(transsetsourcelanguage)setsourcelanguage()(translate.storage.properties.stringsfilemethod),late.storage.jsonl10n.GoI18NJsonFile614		
453 setsourcelanguage() (transsetsourcelanguage() (translate.storage.jsonl10n.GoI18NJsonFile setsourcelanguage() (translate.storage.properties.stringsfile method),		
setsourcelanguage() (trans-late.storage.properties.stringsfile method), late.storage.jsonl10n.GoI18NJsonFile 614		
late.storage.jsonl10n.GoI18NJsonFile 614		

late.storage.properties.stringsutf8file method), 616	setsuggestionstore() (trans- late.filters.checks.CCLicenseChecker method),
setsourcelanguage() (trans-	277
late.storage.properties.xwikifile method),	setsuggestionstore() (trans-
618	late.filters.checks.DrupalChecker method),
setsourcelanguage() (trans-	283
late.storage.properties.XWikiFullPage	setsuggestionstore() (trans-
method), 593	late.filters.checks.GnomeChecker method),
setsourcelanguage() (trans-	288
late.storage.properties.XWikiPageProperties	setsuggestionstore() (trans-
method), 595	late.filters.checks.IOSChecker method), 294
setsourcelanguage() (trans-	setsuggestionstore() (trans-
late.storage.pypo.pofile method), 624	late.filters.checks.KdeChecker method),
setsourcelanguage() (translate.storage.qm.qmfile	300
method), 630	setsuggestionstore() (trans-
	late.filters.checks.L20nChecker method),
· · · · · · · · · · · · · · · · · · ·	
late.storage.qph.QphFile method), 636	306
	setsuggestionstore() (trans-
method), 641	late.filters.checks.LibreOfficeChecker method),
setsourcelanguage() (trans-	311
late. storage. subtitles. Adv SubStation Alpha File	setsuggestionstore() (trans-
method), 648	late.filters.checks.MinimalChecker method),
setsourcelanguage() (trans-	317
late.storage.subtitles.MicroDVDFile method),	setsuggestionstore() (trans-
650	late.filters.checks.MozillaChecker method),
setsourcelanguage() (trans-	323
late.storage.subtitles.SubRipFile method),	setsuggestionstore() (trans-
652	late.filters.checks.OpenOfficeChecker method),
setsourcelanguage() (trans-	329
late.storage.subtitles.SubStationAlphaFile	setsuggestionstore() (trans-
method), 653	late.filters.checks.ReducedChecker method),
	334
setsourcelanguage() (trans-	
late.storage.subtitles.SubtitleFile method),	setsuggestionstore() (trans-
655	late.filters.checks.StandardChecker method),
setsourcelanguage() (translate.storage.tbx.tbxfile	340
method), 660	setsuggestionstore() (trans-
setsourcelanguage() (trans-	late.filters.checks.StandardUnitChecker
late.storage.tiki.TikiStore method), 666	method), 343
setsourcelanguage() (trans-	setsuggestionstore() (trans-
late.storage.tmx.tmxfile method), 671	late.filters.checks.TeeChecker method), 344
setsourcelanguage() (trans-	setsuggestionstore() (trans-
late.storage.trados.TradosTxtTmFile method),	late.filters.checks.TermChecker method),
680	347
<pre>setsourcelanguage() (translate.storage.ts2.tsfile</pre>	setsuggestionstore() (trans-
method), 682	late.filters.checks.TranslationChecker method),
setsourcelanguage() (trans-	350
late.storage.txt.TxtFile method), 688	setsuggestionstore() (trans-
	late.filters.checks.UnitChecker method),
setsourcelanguage() (trans-	350
late.storage.utx.UtxFile method), 693	
setsourcelanguage() (trans-	settarget() (translate.storage.lisa.LISAunit
late.storage.wordfast.WordfastTMFile method),	method), 487
700	settarget() (translate.storage.poxliff.PoXliffUnit
setsourcelanguage() (trans-	method), 583
late.storage.xliff.xlifffile method), 707	settarget() (translate.storage.qph.QphUnit

method), 639	settargetlanguage() (trans-
${\it settarget()} \ \ \textit{(translate.storage.tbx.tbxunit method)} \\ 664$), late.storage.mozilla_lang.LangStore method), 495
settarget() (translate.storage.tmx.tmxunit method 674	$late. storage. omegat. OmegaTFile \\ method),$
settarget() (translate.storage.ts2.tsunit method 686), 500 settargetlanguage() (trans-
${\tt settarget()} \ \textit{(translate.storage.xliff.xliffunit method)} \\ 710$	
settargetlanguage() (tran.	· · · · · · · · · · · · · · · · · · ·
late.storage.base.DictStore method), 405	late.storage.php.LaravelPHPFile method),
settargetlanguage() (tran.	
late.storage.base.TranslationStore method 410	late.storage.php.phpfile method), 564
settargetlanguage() (tran.	
late.storage.catkeys.CatkeysFile method	
417	settargetlanguage() (trans-
settargetlanguage() (tran. late.storage.catkeys.CatkeysHeader method), 575
417	settargetlanguage() (trans-
settargetlanguage() (tran.	0 1 00 00
late.storage.csvl10n.csvfile method), 422	settargetlanguage() (trans-
settargetlanguage() (translate.storage.dtd.dtdfi. method), 429	
settargetlanguage() (tran.	settargetlanguage() (trans- late.storage.properties.javafile method),
late.storage.html.htmlfile method), 438	599
settargetlanguage() (tran.	
late.storage.html.POHTMLParser method	
settargetlanguage() (tran.	
late.storage.ical.icalfile method), 443 settargetlanguage() (translate.storage.ini.inifile	$late. storage. properties. java utf 8 file \\ method),$
method), 448	settargetlanguage() (trans-
settargetlanguage() (tran.	
late.storage.jsonl10n.ARBJsonFile method	
454	settargetlanguage() (trans-
settargetlanguage() (tran. late.storage.jsonl10n.GoI18NJsonFile	
method), 458	settargetlanguage() (trans-
settargetlanguage() (tran.	
late.storage.jsonl10n.I18NextFile method	
463	settargetlanguage() (trans-
settargetlanguage() (translate.storage.jsonl10n.JsonFile method), 468	
settargetlanguage() (tran.	s- settargetlanguage() (trans-
late.storage.jsonl10n.JsonNestedFile method	
settargetlanguage() (tran.	
late.storage.jsonl10n.WebExtensionJsonFile method), 478	late.storage.properties.XWikiFullPage method), 593
settargetlanguage() (tran.	
late.storage.lisa.LISAfile method), 483 settargetlanguage() (translate.storage.mo.mofile	late. storage. properties. XWiki Page Properties
method), 490	settargetlanguage() (trans-

late.storage.pypo.pofile method), 624	method), 261
settargetlanguage() (translate.storage.qm.qmfile	e settimestampoption() (trans-
method), 630	late. convert. po 2 word fast. Wf Option Parser
settargetlanguage() (trans-	<i>method</i>), 266
late.storage.qph.QphFile method), 636	settypecomment() (translate.storage.pypo.pounit
<pre>settargetlanguage() (translate.storage.rc.rcfile method), 641</pre>	e method),627 setXMLlang() (in module trans-
settargetlanguage() (trans-	- · · · · · · · · · · · · · · · · · · ·
late.storage.subtitles.AdvSubStationAlphaFile method), 648	setXMLspace() (in module trans- late.misc.xml_helpers), 400
settargetlanguage() (trans-	
late.storage.subtitles.MicroDVDFile method)	
650	short() (translate.filters.checks.DrupalChecker
settargetlanguage() (trans-	
late.storage.subtitles.SubRipFile method) 652	, short() (translate.filters.checks.GnomeChecker method), 288
settargetlanguage() (trans-	
late.storage.subtitles.SubStationAlphaFile method), 653	294 short() (translate.filters.checks.KdeChecker method),
settargetlanguage() (trans-	
late.storage.subtitles.SubtitleFile method)	
655	method), 306
settargetlanguage() (translate.storage.tbx.tbxfile	
method), 660	method), 311
settargetlanguage() (trans-	
late.storage.tiki.TikiStore method), 666	method), 317
settargetlanguage() (trans-	- short() (translate.filters.checks.MozillaChecker
late.storage.tmx.tmxfile method), 671	method), 323
settargetlanguage() (trans-	
late.storage.trados.TradosTxtTmFile method)	
680	short() (translate.filters.checks.ReducedChecker
settargetlanguage() (translate.storage.ts2.tsfile	
method), 683	short() (translate.filters.checks.StandardChecker
settargetlanguage() (trans-	**
late.storage.txt.TxtFile method), 688	short () (translate.filters.checks.TermChecker method), 347
settargetlanguage() (trans-	
<pre>late.storage.utx.UtxFile method), 693 settargetlanguage() (trans-</pre>	should_output_store() (in module trans- late.convert.convert), 244
late.storage.wordfast.WordfastTMFile method)	
700	method), 398
settargetlanguage() (trans-	
late.storage.xliff.xlifffile method), 707	method), 398
settext() (translate.storage.oo.ooline method), 506	show() (translate.misc.progressbar.MessageProgressBa
settimestampoption() (trans-	
late.convert.convert.ArchiveConvertOptionPars	ser show () (translate.misc.progressbar.NoProgressBar
method), 240	method), 398
settimestampoption() (trans-	- show() (translate.misc.progressbar.ProgressBar
late.convert.convert.ConvertOptionParser	method), 398
method), 244	show() (translate.misc.progressbar.VerboseProgressBar
settimestampoption() (trans-	**
late.convert.po2moz.MozConvertOptionParser	si (class in translate.lang.si), 381
method), 256	simplecaps() (trans-
settimestampoption() (trans-	•
late.convert.po2tmx.TmxOptionParser	277

simplecaps()	(trans-	317	
late.filters.checks.DrupalChecker	method),	simpleplurals()	(trans-
283		late.filters.checks.MozillaChecker	method),
simplecaps()	(trans-	323	
late.filters.checks.GnomeChecker	method),	simpleplurals()	(trans-
289	,,,	late.filters.checks.OpenOfficeChecker	
simplecaps() (translate.filters.checks.IC	SChecker	329	,,,
method), 294	50	simpleplurals()	(trans-
simplecaps() (translate.filters.checks.Ka	leChecker	late.filters.checks.ReducedChecker	method),
method), 300	icenci	335	memou),
simplecaps() (translate.filters.checks.L20	nChecker	simpleplurals()	(trans-
method), 306	meneeker	late.filters.checks.StandardChecker	method),
simplecaps()	(trans-	340	memou),
		simpleplurals()	(trans
late.filters.checks.LibreOfficeChecker 312	memoa),		(trans-
	(4	late.filters.checks.TermChecker	method),
simplecaps()	(trans-	347	
late.filters.checks.MinimalChecker	method),	simplercode() (in module translate.lang.da	
317	()	simplify_to_common() (in module	trans-
simplecaps()	(trans-	late.lang.data), 365	,
late.filters.checks.MozillaChecker	method),	singlequoting()	(trans-
323		late.filters.checks.CCLicenseChecker	method),
simplecaps()	(trans-	277	
late.filters.checks.OpenOfficeChecker	method),	singlequoting()	(trans-
329		late. filters. checks. Drupal Checker	method),
simplecaps()	(trans-	283	
late.filters.checks.ReducedChecker	method),	singlequoting()	(trans-
334		late.filters.checks.GnomeChecker	method),
simplecaps()	(trans-	289	
late.filters.checks.StandardChecker	method),	singlequoting()	(trans-
340		late.filters.checks.IOSChecker method), 295
simplecaps() (translate.filters.checks.Ter	mChecker	singlequoting()	(trans-
method), 347		late.filters.checks.KdeChecker	method),
simpleplurals()	(trans-	300	,,
late.filters.checks.CCLicenseChecker	method).	singlequoting()	(trans-
277	,,	late.filters.checks.L20nChecker	method),
simpleplurals()	(trans-	306	,,
late.filters.checks.DrupalChecker	method),	singlequoting()	(trans-
283	memou),	late.filters.checks.LibreOfficeChecker	
simpleplurals()	(trans-	312	memou),
late.filters.checks.GnomeChecker	method),	singlequoting()	(trans-
289	memou),		method),
	(tuana	late.filters.checks.MinimalChecker	meinoa),
simpleplurals()	(trans-	318	(4
late.filters.checks.IOSChecker method		singlequoting()	(trans-
simpleplurals()	(trans-	late.filters.checks.MozillaChecker	method),
late.filters.checks.KdeChecker	method),	323	7.
300		singlequoting()	(trans-
simpleplurals()	(trans-	late.filters.checks.OpenOfficeChecker	method),
late.filters.checks.L20nChecker	method),	329	
306		singlequoting()	(trans-
simpleplurals()	(trans-	late.filters.checks.ReducedChecker	method),
late. filters. checks. Libre Of fice Checker	method),	335	
312		singlequoting()	(trans-
simpleplurals()	(trans-	late.filters.checks.StandardChecker	method),
late.filters.checks.MinimalChecker	method),	341	

3 1 3 1	(trans- ethod),	spellcheck() (trans- late.filters.checks.GnomeChecker method), 289
source (translate.storage.dtd.dtdunit attribute), 4 source (translate.storage.pypo.pounit attribute),		spellcheck() (translate.filters.checks.IOSChecker method), 295
late.storage.statistics.Statistics me	(trans- ethod),	spellcheck() (translate.filters.checks.KdeChecker method), 300
645 sourcefiles	(trans-	spellcheck() (translate.filters.checks.L20nChecker method), 306
late.storage.bundleprojstore.BundleProje	`	
attribute), 414		late.filters.checks.LibreOfficeChecker method),
sourcefiles (translate.storage.projstore.Projec	ctStore	312
attribute), 585	\ 40 2	spellcheck() (trans-
sourcelen() (in module translate.search.match	,	late.filters.checks.MinimalChecker method),
SourceStoreClass late.convert.ical2po.ical2po attribute), 2-	(trans- 46	318 spellcheck() (trans-
SourceStoreClass (translate.convert.ini2po.		late.filters.checks.MozillaChecker method),
attribute), 247	<i>F</i> =	324
SourceStoreClass	(trans-	spellcheck() (trans-
late.convert.mozlang2po.lang2po attr 248	ribute),	late.filters.checks.OpenOfficeChecker method), 329
	(trans-	spellcheck() (trans-
late.convert.php2po.php2po attribute), 25	,	late.filters.checks.ReducedChecker method),
	(trans-	335 spellcheck() (trans-
late.convert.po2ical.po2ical attribute), 23 SourceStoreClass (translate.convert.po2ini.		late.filters.checks.StandardChecker method),
attribute), 251		341
	(trans-	spellcheck() (translate.filters.checks.TermChecker
	ribute),	method), 348
252	2	split() (translate.misc.multistring.multistring
SourceStoreClass (translate.convert.po2tiki.pattribute), 257	002τικι	method), 393 splitext() (translate.convert.convert.ArchiveConvertOptionParser
	(trans-	method), 240
	ibute),	splitext() (translate.convert.convert.ConvertOptionParser
267	,,	method), 244
SourceStoreClass (translate.convert.tiki2po.tatribute), 270	tiki2po	<pre>splitext() (translate.convert.po2moz.MozConvertOptionParser</pre>
SourceStoreClass (translate.convert.txt2po. attribute), 271	.txt2po	<pre>splitext() (translate.convert.po2tmx.TmxOptionParser method), 261</pre>
	(trans-	splitext() (translate.convert.po2wordfast.WfOptionParser
* * * *	ribute),	method), 266
272	ation)	splitext() (translate.filters.pofilter.FilterOptionParser method), 355
spaceend() (in module translate.filters.decor	anon),	splitext() (translate.misc.optrecurse.RecursiveOptionParser
spacestart() (in module	trans-	method), 396
late.filters.decoration), 352		splitext() (translate.tools.poconflicts.ConflictOptionParser
specialchars (translate.lang.common.Commo	on at-	method), 718
tribute), 363		<pre>splitext() (translate.tools.pogrep.GrepOptionParser</pre>
	(trans-	method), 722
late.filters.checks.CCLicenseChecker me	ethod),	splitext() (translate.tools.porestructure.SplitOptionParser
277 spellcheck()	(trans-	method), 725 splitext() (translate.tools.poterminology.TerminologyOptionParse
	(trans- ethod),	method), 729
283	,	splitinputext() (trans-

late.convert.convert.ArchiveConvertOptionParse method), 240	er late.filters.pofilter.FilterOptionParser method), 355
splitinputext() (trans- late.convert.convert.ConvertOptionParser method), 244	<pre>splittemplateext() (trans- late.misc.optrecurse.RecursiveOptionParser method), 396</pre>
splitinputext() (trans- late.convert.po2moz.MozConvertOptionParser method), 256	<pre>splittemplateext() (trans- late.tools.poconflicts.ConflictOptionParser method), 718</pre>
<pre>splitinputext() (trans- late.convert.po2tmx.TmxOptionParser</pre>	<pre>splittemplateext() (trans- late.tools.pogrep.GrepOptionParser method),</pre>
<pre>method), 261 splitinputext()</pre>	722 splittemplateext() (trans-
late.convert.po2wordfast.WfOptionParser method), 266	late.tools.porestructure.SplitOptionParser method), 725
splitinputext() (trans- late.filters.pofilter.FilterOptionParser method), 355	splittemplateext() (trans- late.tools.poterminology.TerminologyOptionParser method), 729
splitinputext() (trans- late.misc.optrecurse.RecursiveOptionParser method), 396	st (class in translate.lang.st), 381 StandardChecker (class in translate.filters.checks), 337
splitinputext() (trans- late.tools.poconflicts.ConflictOptionParser	StandardUnitChecker (class in trans- late.filters.checks), 342
<pre>method), 718 splitinputext() (trans- late.tools.pogrep.GrepOptionParser method),</pre>	start_namespace_decl_handler() (trans- late.misc.ourdom.ExpatBuilderNS method), 397
722 splitinputext() (trans- late.tools.porestructure.SplitOptionParser	startcaps() (trans- late.filters.checks.CCLicenseChecker method), 277
<pre>method), 725 splitinputext() (trans-</pre>	startcaps() (translate.filters.checks.DrupalChecker method), 283
method), 729	sestartcaps () (translate.filters.checks.GnomeChecker method), 289
splitlines() (in module translate.storage.pypo), 628	method), 295
splitlines() (translate.misc.multistring.multistring method), 393	startcaps() (translate.filters.checks.KdeChecker method), 301
SplitOptionParser (class in trans- late.tools.porestructure), 723	startcaps() (translate.filters.checks.L20nChecker method), 307
splittemplateext() (trans- late.convert.convert.ArchiveConvertOptionParse method), 240	startcaps() (transer late.filters.checks.LibreOfficeChecker method), 312
<pre>splittemplateext() (trans- late.convert.convert.ConvertOptionParser method), 244</pre>	startcaps() (trans- late.filters.checks.MinimalChecker method), 318
<pre>splittemplateext() (trans- late.convert.po2moz.MozConvertOptionParser method), 256</pre>	startcaps() (translate.filters.checks.MozillaChecker method), 324 startcaps() (trans-
<pre>splittemplateext()</pre>	late.filters.checks.OpenOfficeChecker method), 329
<pre>method), 261 splittemplateext() (trans- late.convert.po2wordfast.WfOptionParser</pre>	startcaps() (trans- late.filters.checks.ReducedChecker method), 335
<pre>method), 266 splittemplateext() (trans-</pre>	startcaps() (trans- late.filters.checks.StandardChecker method),

341	startwhitespace() (trans-
startcaps() (translate.filters.checks.TermChecker method), 348	late.filters.checks.LibreOfficeChecker method), 313
startpunc() (trans-	startwhitespace() (trans-
late.filters.checks.CCLicenseChecker method), 278	late.filters.checks.MinimalChecker method), 318
$\verb startpunc() \textit{(translate.filters.checks.Drupal Checker} $	startwhitespace() (trans-
method), 284	late.filters.checks.MozillaChecker method),
$\verb startpunc() \textit{(translate.filters.checks.GnomeChecker)} $	324
method), 289	startwhitespace() (trans-
startpunc() (translate.filters.checks.IOSChecker method), 295	late.filters.checks.OpenOfficeChecker method), 330
startpunc() (translate.filters.checks.KdeChecker	startwhitespace() (trans-
method), 301	late.filters.checks.ReducedChecker method),
startpunc() (translate.filters.checks.L20nChecker	335
method), 307	startwhitespace() (trans-
startpunc() (trans-	late.filters.checks.StandardChecker method),
late.filters.checks.LibreOfficeChecker method),	341
312	startwhitespace() (trans-
startpunc() (trans-	late.filters.checks.TermChecker method),
late.filters.checks.MinimalChecker method),	348
318	StateEnum (class in translate.storage.workflow), 704
startpunc() (translate.filters.checks.MozillaChecker	
method), 324	646
startpunc() (trans-	statemap (translate.storage.ts2.tsunit attribute), 686
late.filters.checks.OpenOfficeChecker method),	StateNotInWorkflowError, 704
330	Statistics (class in translate.storage.statistics), 645
startpunc() (trans-	StatsCache (class in translate.storage.statsdb), 646
late.filters.checks.ReducedChecker method),	str2bool() (in module translate.tools.pomerge), 722
335	string_xpath (in module trans-
startpunc() (trans-	late.misc.xml_helpers), 400
late.filters.checks.StandardChecker method),	
341	late.misc.xml_helpers), 400
startpunc() (translate.filters.checks.TermChecker	• •
method), 348	late.storage.placeables.strelem), 537
startswith() (translate.misc.multistring.multistring	
method), 393	613
	stringsutf8file (class in trans-
late.filters.checks.CCLicenseChecker method),	late.storage.properties), 614
278	strip() (translate.misc.multistring.multistring
startwhitespace() (trans-	method), 393
late.filters.checks.DrupalChecker method),	strip_missing_part() (trans-
284	late.storage.properties.propunit class method),
startwhitespace() (trans-	612
late.filters.checks.GnomeChecker method),	strip_missing_part() (trans-
290	late.storage.properties.xwikiunit class method),
startwhitespace() (trans-	621
late.filters.checks.IOSChecker method), 295	Sub (class in translate.storage.placeables.base), 521
startwhitespace() (trans-	Sub (class in translate.storage.placeables.xliff), 552
late.filters.checks.KdeChecker method),	sub (translate.storage.placeables.strelem.StringElem at-
301	tribute), 539
startwhitespace() (trans-	SubflowPlaceable (class in trans-
late.filters.checks.L20nChecker method),	late.storage.placeables.interfaces), 535
307	SubRipFile (class in translate.storage.subtitles), 650

SubStationAlphaFile (class in trans- late.storage.subtitles), 652	tagproperties() (in module trans- late.filters.checks), 351
SubtitleFile (class in translate.storage.subtitles). 654	$late.misc.optrecurse. Man Page Option\ method),$
SubtitleUnit (class in translate.storage.subtitles)	
655 suggestions_in_format (trans-	target (translate.storage.dtd.dtdunit attribute), 432 target (translate.storage.pypo.pounit attribute), 628
late.storage.base.TranslationStore attribute).	
410	targetfiles (trans-
suggestions_in_format (trans- late.storage.xliff.xlifffile attribute), 707	
summarize() (in module translate.tools.pocount), 718	
<pre>supported_files() (in module trans- late.storage.factory), 433</pre>	
sv (class in translate.lang.sv), 382	late.convert.ical2po.ical2po attribute), 246
swapcase() (translate.misc.multistring.multistring method), 393	
swapdir() (in module translate.tools.poswap), 726	TargetStoreClass (trans-
<pre>switchfile() (translate.storage.poxliff.PoXliffFile</pre>	late.convert.mozlang2po.lang2po attribute), 248
$\verb switchfile() (translate.storage.xliff.xlifffile method) .\\$	TargetStoreClass (trans-
707	late.convert.php2po.php2po attribute), 250
Т	TargetStoreClass (trans-
	late.convert.po2ical.po2ical attribute), 251
ta (class in translate.lang.ta), 383	TargetStoreClass (translate.convert.po2ini.po2ini
TAB_UTF16 (in module translate.storage.wordfast), 698	
tabs () (translate.filters.checks.CCLicenseChecker	TargetStoreClass (trans- late.convert.po2mozlang.po2lang attribute),
method), 278 tabs () (translate.filters.checks.DrupalChecker	
method), 284	TargetStoreClass (translate.convert.po2tiki.po2tiki
tabs () (translate.filters.checks.GnomeChecker	
method), 290	TargetStoreClass (trans-
tabs () (translate.filters.checks.IOSChecker method).	
tabs() (translate.filters.checks.KdeChecker method).	TargetStoreClass (translate.convert.tiki2po.tiki2po attribute), 270
tabs() (translate.filters.checks.L20nChecker method). 307	TargetStoreClass (translate.convert.txt2po.txt2po attribute), 271
${\tt tabs} \ () \ \ \ \ (\textit{translate.filters.checks.LibreOfficeChecker})$	
method), 313	late.convert.yaml2po.yaml2po attribute),
tabs() (translate.filters.checks.MinimalChecker	
method), 318	TargetUnitClass (translate.convert.ical2po.ical2po
tabs() (translate.filters.checks.MozillaChecker	
method), 324	TargetUnitClass (translate.convert.ini2po.ini2po attribute), 247
tabs () (translate.filters.checks.OpenOfficeChecker method), 330	TargetUnitClass (trans-
tabs () (translate.filters.checks.ReducedChecker	
method), 335	248
tabs () (translate.filters.checks.StandardChecker	
method), 341	attribute), 250
tabs() (translate.filters.checks.TermChecker method). 348	
tagname () (in module translate.filters.checks), 351	<pre>TargetUnitClass (translate.convert.po2ini.po2ini</pre>

TargetUnitClass	(trans-	late.search.match), 402
late.convert.po2mozlang.po2lang 252	attribute),	TerminologyOptionParser (class in trans- late.tools.poterminology), 726
<pre>TargetUnitClass (translate.convert.po</pre>	2tiki.po2tiki	TerminologyPlaceable (class in trans- late.storage.placeables.terminology), 539
TargetUnitClass	(trans-	text (translate.storage.oo.ooline attribute), 506
late.convert.po2yaml.po2yaml	attribute),	th (class in translate.lang.th), 384
267	,,	tiki2po (class in translate.convert.tiki2po), 270
TargetUnitClass (translate.convert.tik	i2no tiki2no	TikiStore (class in translate.storage.tiki), 664
attribute), 270	игролимгро	TikiUnit (class in translate.storage.tiki), 666
TargetUnitClass (translate.convert.t.	xt2no txt2no	
attribute), 271	м2роли2ро	675
TargetUnitClass	(trans-	time (translate.storage.wordfast.WordfastTime at-
late.convert.yaml2po.yaml2po	attribute),	tribute), 700
272	annome,	timestring (translate.storage.trados.TradosTxtDate
	0	attribute), 675
tbxfile (class in translate.storage.tbx), 65		
tbxunit (class in translate.storage.tbx), 66	1	timestring (translate.storage.wordfast.WordfastTime
te (class in translate.lang.te), 384		attribute), 700
TeeChecker (class in translate.filters.check		title() (translate.misc.multistring.multistring
<pre>templateexists()</pre>	(trans-	method), 393
	tOptionParse	er TMServer (class in translate.services.tmserver), 403
method), 240		tmxfile (class in translate.storage.tmx), 669
<pre>templateexists()</pre>	(trans-	TmxOptionParser (class in trans-
late.convert.convert.ConvertOption	Parser	late.convert.po2tmx), 258
method), 244		tmxunit (class in translate.storage.tmx), 671
<pre>templateexists()</pre>	(trans-	tr_lang() (in module translate.lang.data), 365
late.convert.po2moz.MozConvertOp	otionParser	TRADOS_TIMEFORMAT (in module trans-
method), 256		late.storage.trados), 675
<pre>templateexists()</pre>	(trans-	TradosTxtDate (class in translate.storage.trados),
late.convert.po2tmx.TmxOptionPar.	ser	675
method), 261		<pre>TradosTxtTmFile (class in translate.storage.trados),</pre>
<pre>templateexists()</pre>	(trans-	679
late.convert.po2wordfast.WfOption.	Parser	TradosUnit (class in translate.storage.trados), 676
method), 266		tranliterate_cyrillic() (in module trans-
templateexists()	(trans-	late.lang.af), 357
late.filters.pofilter.FilterOptionPars	,	
355	,,,	646
templateexists()	(trans-	transfiles (translate.storage.bundleprojstore.BundleProjectStorage)
late.misc.optrecurse.RecursiveOptic		attribute), 415
method), 396	one corser	transfiles (translate.storage.projstore.ProjectStore
templateexists()	(trans-	attribute), 585
late.tools.poconflicts.ConflictOption	`	TransitionError, 704
method), 718	ii arser	Translatable (class in trans-
templateexists()	(trans-	late.storage.xml_extract.extract), 711
late.tools.pogrep.GrepOptionParser	`	
	meinoa),	
722	(,	late.storage.html.htmlfile attribute), 435
templateexists()	(trans-	TRANSLATABLE_ELEMENTS (trans-
late.tools.porestructure.SplitOption	rarser	late.storage.html.htmlfile attribute), 435
method), 725		TRANSLATABLE_METADATA (trans-
<pre>templateexists()</pre>	(trans-	late.storage.html.htmlfile attribute), 435
method), 729		setranslate() (translate.misc.multistring.multistring method), 393
TermChecker (class in translate.filters.che	cks), 344	translate() (translate.storage.base.DictStore
terminologymatcher (class in	ı trans-	method), 405

translate() (translate.storage.base.TranslationStore translate() (translate.storage.placeables.base.Ex method), 519 *method*), 410 translate() (translate.storage.catkeys.CatkeysFile translate() (translate.storage.placeables.base.G method), 516 method), 417 translate() (translate.storage.csvl10n.csvfile translate() (translate.storage.placeables.base.It method), 422 method), 514 translate() (translate.storage.dtd.dtdfile method), (translate.storage.placeables.base.Ph translate() 429 method), 513 translate() (translate.storage.html.htmlfile method), translate() (translate.storage.placeables.base.Sub 438 method), 522 translate() (translate.storage.html.POHTMLParser translate() (translate.storage.placeables.base.X *method*), 435 method), 521 translate() (translate.storage.ical.icalfile method), (transtranslate() 443 late.storage.placeables.general.AltAttrPlaceable (translate.storage.ini.inifile method), method), 524 translate() 448 translate() (translate.storage.placeables.general.XMLEntityPlaceable translate() (translate.storage.jsonl10n.ARBJsonFile method), 526 method), 454 (transtranslate() translate() (translate.storage.placeables.general.XMLTagPlaceable late.storage.jsonl10n.GoI18NJsonFile method), 528 *method*), 459 (transtranslate() translate() (translate.storage.jsonl10n.I18NextFile late.storage.placeables.interfaces.BasePlaceable method), 463 method), 529 translate() (translate.storage.jsonl10n.JsonFile translate() (transmethod), 468 late. storage. placeables. interfaces. Invisible Placeable(transmethod), 531 translate() late.storage.jsonl10n.JsonNestedFile method), translate() (trans-470 late.storage.placeables.interfaces.MaskingPlaceable (transmethod), 533 translate() late.storage.jsonl10n.WebExtensionJsonFile translate() (trans*method*), 478 late.storage.placeables.interfaces.ReplacementPlaceable translate() (translate.storage.lisa.LISAfile method), method), 535 483 (transtranslate() translate() (translate.storage.mo.mofile method), late.storage.placeables.interfaces.SubflowPlaceable 490 method), 536 translate() (transtranslate() (translate.storage.mozilla_lang.LangStore method), late.storage.placeables.strelem.StringElem495 method), 539 translate() (translate.storage.omegat.OmegaTFile translate() (translate.storage.placeables.terminology.TerminologyPlaceable *method*), 500 translate() method), 541 (transtranslate() (translate.storage.placeables.xliff.Bpt late.storage.omegat.OmegaTFileTab method), method), 543 translate() (translate.storage.php.LaravelPHPFile translate() (translate.storage.placeables.xliff.Bx method), 560method), 547 translate() (translate.storage.php.phpfile method), translate() (translate.storage.placeables.xliff.Ept 564 method), 544 translate() (translate.storage.placeables.base.Bpt translate() (translate.storage.placeables.xliff.Ex method), 510method), 549 (translate.storage.placeables.xliff.G (translate.storage.placeables.base.Bx translate() translate() *method*), 518 method), 551 translate() (translate.storage.placeables.base.Ept translate() (translate.storage.placeables.xliff.It

Index 861

method), 552

method), 511

translate() (translate.storage.placeables.xliff.Ph method), 555	$translate() \\ late. storage. subtitles. SubStation Alpha File$
translate() (translate.storage.placeables.xliff.Sub	method), 654
method), 554	${\tt translate()} \textit{(translate.storage.subtitles. Subtitle File)}$
translate() (trans-	method), 655
late.storage.placeables.xliff.UnknownXML method), 557	translate() (translate.storage.tbx.tbxfile method), 660
translate() (translate.storage.placeables.xliff.X method), 546	translate() (translate.storage.tiki.TikiStore method), 666
translate() (translate.storage.pocommon.pofile method), 570	translate() (translate.storage.tmx.tmxfile method), 671
translate() (translate.storage.poxliff.PoXliffFile method), 579	<pre>translate() (trans- late.storage.trados.TradosTxtTmFile method),</pre>
translate() (translate.storage.properties.gwtfile	680
method), 597	translate() (translate.storage.ts2.tsfile method), 683
translate() (translate.storage.properties.javafile method), 599	translate() (translate.storage.txt.TxtFile method), 688
translate() (trans- late.storage.properties.javautf16file method),	translate() (translate.storage.utx.UtxFile method), 693
601	translate() (trans-
translate() (translate.storage.properties.javautf8file method), 603	late.storage.wordfast.WordfastTMFile method), 700
translate() (translate.storage.properties.joomlafile method), 605	translate() (translate.storage.xliff.xlifffile method), 707
translate() (translate.storage.properties.propfile	translate.convert (module), 236
method), 606	translate.convert.accesskey (module), 236
translate() (translate.storage.properties.stringsfile	translate.convert.convert (module), 237
method), 614	translate.convert.csv2po (module), 244
translate() (translate.storage.properties.stringsutf8file method),	translate.convert.csv2tbx (module), 245 translate.convert.dtd2po (module), 245
616	translate.convert.factory (module), 246
translate() (translate.storage.properties.xwikifile	translate.convert.html2po(module), 246
method), 618	translate.convert.ical2po (module), 246
translate() (trans-	translate.convert.ini2po (module), 247
late.storage.properties.XWikiFullPage	translate.convert.json2po(module), 247
method), 593	translate.convert.moz2po(<i>module</i>), 248
translate() (trans-	translate.convert.mozfunny2prop (module),
late. storage. properties. XWiki Page Properties	248
method), 595	translate.convert.mozlang2po(module), 248
translate() (translate.storage.pypo.pofile method), 624	translate.convert.odf2xliff(<i>module</i>), 249 translate.convert.oo2po(<i>module</i>), 249
translate() (translate.storage.qm.qmfile method), 630	translate.convert.oo2xliff (module), 249 translate.convert.php2po (module), 250
translate() (translate.storage.qph.QphFile method), 636	translate.convert.po2csv(module), 250 translate.convert.po2dtd(module), 250
translate() (translate.storage.rc.rcfile method), 641	translate.convert.po2html(module),251
translate() (trans-	translate.convert.po2ical(module),251
late.storage.subtitles.AdvSubStationAlphaFile	translate.convert.po2ini(module),251
method), 648	translate.convert.po2json(module), 252
translate() (trans-	translate.convert.po2moz (module), 253
late.storage.subtitles.MicroDVDFile method), 650	translate.convert.po2mozlang(module), 252
translate() (translate.storage.subtitles.SubRipFile	translate.convert.po2oo(module), 256 translate.convert.po2php(module), 256
method), 652	translate.convert.po2pnp (module), 256
	I I I //

```
translate.convert.po2rc(module), 257
                                             translate.lang.he (module), 371
                                             translate.lang.hi (module), 372
translate.convert.po2resx(module), 257
translate.convert.po2sub(module), 257
                                             translate.lang.hy (module), 373
translate.convert.po2symb (module), 257
                                             translate.lang.identify (module), 373
translate.convert.po2tiki(module), 257
                                             translate.lang.ja (module), 373
                                             translate.lang.km (module), 374
translate.convert.po2tmx (module), 258
                                             translate.lang.kn (module), 375
translate.convert.po2ts (module), 262
                                             translate.lang.ko (module), 376
translate.convert.po2txt (module), 262
translate.convert.po2web2py (module), 262
                                             translate.lang.ml (module), 377
translate.convert.po2wordfast
                                    (module),
                                             translate.lang.mr (module), 377
                                             translate.lang.ne (module), 378
                                             translate.lang.ngram (module), 379
translate.convert.po2xliff (module), 266
translate.convert.po2yaml (module), 266
                                             translate.lang.pa (module), 379
                                             translate.lang.poedit (module), 380
translate.convert.pot2po (module), 267
                                             translate.lang.si(module),380
translate.convert.prop2mozfunny (module),
       267
                                             translate.lang.st (module), 381
                                             translate.lang.sv(module), 382
translate.convert.prop2po (module), 268
translate.convert.rc2po (module), 269
                                             translate.lang.ta(module), 383
translate.convert.resx2po (module), 269
                                             translate.lang.te(module), 384
translate.convert.sub2po (module), 269
                                             translate.lang.team (module), 383
translate.convert.symb2po (module), 270
                                             translate.lang.th (module), 384
translate.convert.tiki2po (module), 270
                                             translate.lang.ug (module), 385
translate.convert.ts2po (module), 270
                                             translate.lang.ur (module), 386
                                             translate.lang.vi(module).387
translate.convert.txt2po (module), 271
translate.convert.web2py2po (module), 271
                                             translate.lang.zh (module), 388
translate.convert.xliff2odf (module), 271
                                             translate.misc (module), 388
translate.convert.xliff2oo (module), 272
                                             translate.misc.dictutils (module), 388
translate.convert.xliff2po (module), 272
                                             translate.misc.file_discovery
                                                                                  (module),
translate.convert.yaml2po(module), 272
translate.filters (module), 272
                                             translate.misc.multistring (module), 389
translate.filters.autocorrect
                                    (module),
                                             translate.misc.optrecurse (module), 393
       273
                                             translate.misc.ourdom(module), 397
translate.filters.checks (module), 273
                                             translate.misc.progressbar (module), 398
translate.filters.decoration(module),351
                                             translate.misc.quote (module), 399
translate.filters.helpers (module), 352
                                             translate.misc.wsgi (module), 400
translate.filters.pofilter (module), 352
                                             translate.misc.xml_helpers(module), 400
translate.filters.prefilters (module), 355
                                             translate.search (module), 401
translate.filters.spelling (module), 356
                                             translate.search.lshtein (module), 401
translate.lang (module), 356
                                             translate.search.match (module), 401
translate.lang.af (module), 357
                                             translate.search.terminology (module), 403
                                             translate.services (module), 403
translate.lang.am (module), 358
translate.lang.ar (module), 358
                                             translate.services.tmserver (module), 403
translate.lang.bn (module), 359
                                             translate.storage (module), 403
translate.lang.code_or(module), 360
                                             translate.storage._factory_classes(mod-
translate.lang.common (module), 361
                                                     ule), 432
translate.lang.data(module), 364
                                             translate.storage.base (module), 403
translate.lang.de (module), 365
                                             translate.storage.benchmark (module), 414
translate.lang.el (module), 366
                                             translate.storage.bundleprojstore (mod-
translate.lang.es (module), 367
                                                     ule), 414
translate.lang.fa (module), 368
                                             translate.storage.catkeys (module), 415
translate.lang.factory (module), 368
                                             translate.storage.csv110n(module),421
translate.lang.fi(module), 369
                                             translate.storage.directory (module), 426
translate.lang.fr (module), 369
                                             translate.storage.dtd(module), 426
translate.lang.gu (module), 370
                                             translate.storage.factory (module), 432
```

```
translate.storage.html (module), 433
                                              translate.storage.utx (module), 691
translate.storage.ical (module), 441
                                              translate.storage.wordfast (module), 697
translate.storage.ini(module), 446
                                              translate.storage.workflow(module), 703
translate.storage.jsonl10n (module), 451
                                              translate.storage.xliff (module), 704
translate.storage.lisa(module), 482
                                              translate.storage.xml_extract
                                                                                  (module),
translate.storage.mo (module), 487
                                                     711
translate.storage.mozilla lang (module),
                                              translate.storage.xml extract.extract
                                                      (module), 711
translate.storage.odf_io(module), 498
                                              translate.storage.xml_extract.generate
translate.storage.odf_shared(module), 498
                                                     (module), 711
translate.storage.omegat (module), 498
                                              translate.storage.xml_extract.misc(mod-
translate.storage.oo(module), 505
                                                     ule), 712
translate.storage.php (module), 557
                                              translate.storage.xml_extract.unit_tree
                                                      (module), 712
translate.storage.placeables (module), 508
translate.storage.placeables.base (mod-
                                              translate.storage.xml_extract.xpath_breadcrumb
       ule), 508
                                                      (module), 713
translate.storage.placeables.general
                                              translate.storage.xml_name (module), 713
       (module), 523
                                              translate.storage.zip(module),714
translate.storage.placeables.interfaces translate.tools (module), 714
       (module), 528
                                              translate.tools.build tmdb (module), 714
translate.storage.placeables.lisa (mod-
                                              translate.tools.phppo2pypo(module),714
                                              translate.tools.poclean (module), 715
                                              translate.tools.pocompile (module), 715
translate.storage.placeables.parse(mod-
                                              translate.tools.poconflicts(module),715
       ule), 536
translate.storage.placeables.strelem
                                              translate.tools.pocount (module), 718
       (module), 537
                                              translate.tools.podebug (module), 719
translate.storage.placeables.terminologytranslate.tools.pogrep (module), 719
       (module), 539
                                              translate.tools.pomerge (module), 722
translate.storage.placeables.xliff(mod-
                                              translate.tools.porestructure
                                                                                  (module),
       ule), 541
                                                     722
translate.storage.po(module), 575
                                              translate.tools.posegment (module), 725
translate.storage.pocommon (module), 568
                                              translate.tools.poswap(module), 726
translate.storage.poheader (module), 574
                                              translate.tools.poterminology
translate.storage.poparser (module), 575
                                                     726
translate.storage.poxliff (module), 576
                                              translate.tools.pretranslate (module), 729
translate.storage.project (module), 583
                                              translate.tools.pydiff (module), 730
translate.storage.projstore (module), 584
                                              translate.tools.pypo2phppo (module), 730
translate.storage.properties (module), 585
                                              TranslateBenchmarker
                                                                        (class
                                                                                in
                                                                                     trans-
translate.storage.pypo (module), 621
                                                      late.storage.benchmark), 414
translate.storage.qm(module),628
                                              translated_unitcount()
                                                                                     (trans-
translate.storage.gph (module), 633
                                                     late.storage.statistics.Statistics
                                                                                   method),
translate.storage.rc(module), 639
                                                     645
translate.storage.statistics (module), 645
                                              translated units()
                                                                                     (trans-
                                                     late. storage. statistics. Statistics \\
translate.storage.statsdb(module),646
                                                                                   method),
translate.storage.subtitles (module), 646
translate.storage.symbian (module), 659
                                              translated_wordcount()
                                                                                     (trans-
translate.storage.tbx (module), 659
                                                     late.storage.statistics.Statistics
                                                                                   method),
                                                     645
translate.storage.tiki (module), 664
translate.storage.tmdb(module),669
                                              TranslationChecker
                                                                       (class
                                                                               in
                                                                                     trans-
translate.storage.tmx (module), 669
                                                     late.filters.checks), 349
translate.storage.trados (module), 675
                                              translations
                                                                                     (trans-
                                                     late.storage.placeables.terminology.TerminologyPlaceable
translate.storage.ts (module), 686
translate.storage.ts2 (module), 680
                                                     attribute), 541
translate.storage.txt (module), 686
                                              TranslationStore (class in translate.storage.base),
```

408	unit_iter() (translate.storage.base.DictStore
TranslationUnit (class in translate.storage.base), 410	<pre>method), 405 unit_iter() (translate.storage.base.DictUnit</pre>
tsfile (class in translate.storage.ts2), 681	method), 408
tsunit (class in translate.storage.ts2), 683 txt2po (class in translate.convert.txt2po), 271	unit_iter() (translate.storage.base.TranslationStore method), 410
TxtFile (class in translate.storage.txt), 686	<pre>unit_iter() (translate.storage.base.TranslationUnit</pre>
TxtUnit (class in translate.storage.txt), 688	method), 413
tzstring() (in module translate.storage.poheader), 575	unit_iter() (translate.storage.catkeys.CatkeysFile method), 417
U	unit_iter() (translate.storage.catkeys.CatkeysUnit method), 420
ug (class in translate.lang.ug), 385	<pre>unit_iter() (translate.storage.csvl10n.csvfile</pre>
unchanged() (trans-	method), 422
late.filters.checks.CCLicenseChecker method),	unit_iter() (translate.storage.csvl10n.csvunit
278	method), 425
unchanged() (translate.filters.checks.DrupalChecker	<pre>unit_iter() (translate.storage.directory.Directory</pre>
method), 284	method), 426
unchanged() (translate.filters.checks.GnomeChecker method), 290	<pre>unit_iter() (translate.storage.dtd.dtdfile method),</pre>
unchanged() (translate.filters.checks.IOSChecker	<pre>unit_iter() (translate.storage.dtd.dtdunit method),</pre>
method), 295	432
unchanged() (translate.filters.checks.KdeChecker method), 301	unit_iter() (translate.storage.html.htmlfile method), 438
unchanged() (translate.filters.checks.L20nChecker	unit_iter() (translate.storage.html.htmlunit
method), 307	method), 441
unchanged() (trans-	<pre>unit_iter() (translate.storage.html.POHTMLParser</pre>
late.filters.checks.LibreOfficeChecker method),	method), 435
313	<pre>unit_iter() (translate.storage.ical.icalfile method),</pre>
unchanged() (trans-	443
late.filters.checks.MinimalChecker method), 318	unit_iter() (translate.storage.ical.icalunit method), 446
unchanged() (translate.filters.checks.MozillaChecker method), 324	unit_iter() (translate.storage.ini.inifile method), 448
unchanged() (trans-	<pre>unit_iter() (translate.storage.ini.iniunit method),</pre>
late.filters.checks.OpenOfficeChecker method),	451
330	unit_iter() (trans-
unchanged() (trans-	late.storage.jsonl10n.ARBJsonFile method),
late.filters.checks.ReducedChecker method),	454
336	unit_iter() (trans-
unchanged() (trans-	late.storage.jsonl10n.ARBJsonUnit method),
late.filters.checks.StandardChecker method),	457
341	unit_iter() (trans-
unchanged() (translate.filters.checks.TermChecker method), 348	late.storage.jsonl10n.GoI18NJsonFile method), 459
unescape() (in module translate.storage.pypo), 628	unit_iter() (trans-
unescape () (in module translate.storage.trados), 675	late.storage.jsonl10n.GoI18NJsonUnit
unescape_help_text() (in module trans-	method), 462
late.storage.oo), 507	<pre>unit_iter() (translate.storage.jsonl10n.I18NextFile</pre>
unescape_text() (in module translate.storage.oo),	method), 464
507	<pre>unit_iter() (translate.storage.jsonl10n.I18NextUnit</pre>
unified_diff() (translate.tools.pydiff.FileDiffer	method), 467
method), 730	unit_iter() (translate.storage.jsonl10n.JsonFile
uni+2dic+() (in module translate search match) 403	method). 468

<pre>unit_iter() (trans-</pre>	method), 599
late.storage.jsonl10n.JsonNestedFile method),	unit_iter() (trans-
470	late.storage.properties.javautf16file method), 601
unit_iter() (trans-	
late.storage.jsonl10n.JsonNestedUnit method), 473	unit_iter() (translate.storage.properties.javautf8file method), 603
<pre>unit_iter() (translate.storage.jsonl10n.JsonUnit</pre>	unit_iter() (translate.storage.properties.joomlafile method), 605
unit_iter() (trans-	<pre>unit_iter() (translate.storage.properties.propfile</pre>
late.storage.jsonl10n.WebExtensionJsonFile method), 478	method), 607 unit_iter() (trans-
<pre>unit_iter()</pre>	late.storage.properties.proppluralunit method), 609
method), 481	<pre>unit_iter() (translate.storage.properties.propunit</pre>
<pre>unit_iter() (translate.storage.lisa.LISAfile method),</pre>	method), 613
484 (tuguelate atongos lieg LISA unit	unit_iter() (translate.storage.properties.stringsfile
unit_iter() (translate.storage.lisa.LISAunit method), 487	method), 614 unit_iter() (trans-
unit_iter() (translate.storage.mo.mofile method), 490	late.storage.properties.stringsutf8file method), 616
unit_iter() (translate.storage.mo.mounit method), 493	unit_iter() (translate.storage.properties.xwikifile method), 618
<pre>unit_iter() (trans-</pre>	unit_iter() (trans-
late.storage.mozilla_lang.LangStore method), 495	late.storage.properties.XWikiFullPage method), 593
unit_iter() (trans-	unit_iter() (trans-
late.storage.mozilla_lang.LangUnit method), 498	late.storage.properties.XWikiPageProperties method), 595
	unit_iter() (translate.storage.properties.xwikiunit method), 621
unit_iter() (trans-	<pre>unit_iter() (translate.storage.pypo.pofile method),</pre>
late.storage.omegat.OmegaTFileTab method),	624
502 unit_iter() (translate.storage.omegat.OmegaTUnit	unit_iter() (translate.storage.pypo.pounit method), 628
method), 505	unit_iter() (translate.storage.qm.qmfile method),
<pre>unit_iter() (translate.storage.php.LaravelPHPFile</pre>	630
method), 560	<pre>unit_iter() (translate.storage.qm.qmunit method),</pre>
<pre>unit_iter() (translate.storage.php.LaravelPHPUnit</pre>	633 unit_iter() (translate.storage.qph.QphFile method),
unit_iter() (translate.storage.php.phpfile method),	636
565	unit_iter() (translate.storage.qph.QphUnit
<pre>unit_iter() (translate.storage.php.phpunit method),</pre>	method), 639
567 unit_iter() (translate.storage.pocommon.pofile	unit_iter() (translate.storage.rc.rcfile method), 642 unit_iter() (translate.storage.rc.rcunit method), 644
method), 570	unit_iter() (translate.storage.rc.rctata memoa), 044
unit_iter() (translate.storage.pocommon.pounit	late.storage.subtitles.AdvSubStationAlphaFile
method), 573	method), 648
<pre>unit_iter() (translate.storage.poxliff.PoXliffFile</pre>	unit_iter() (trans-
method), 579	$late. storage. subtitles. {\it MicroDVDFile} method),$
unit_iter() (translate.storage.poxliff.PoXliffUnit	650
method), 583	unit_iter() (translate.storage.subtitles.SubRipFile
unit_iter() (translate.storage.properties.gwtfile	method), 652
<pre>method), 597 unit_iter() (translate.storage.properties.javafile</pre>	unit_iter() (trans-
<pre>unit_iter() (translate.storage.properties.javafile</pre>	late. storage. subtitles. SubStation Alpha File

method), 654 unit iter() (translate.storage.subtitles.SubtitleFile *method*), 655 unit_iter() (translate.storage.subtitles.SubtitleUnit *method*), 658 unit iter() (translate.storage.tbx.tbxfile method), 441 unit iter() (translate.storage.tbx.tbxunit method), 664 UnitClass unit_iter() (translate.storage.tiki.TikiStore method), unit_iter() (translate.storage.tiki.TikiUnit method), unit_iter() (translate.storage.tmx.tmxfile method), 671 unit_iter() (translate.storage.tmx.tmxunit method), 675 unit_iter() (translate.storage.trados.TradosTxtTmFile method), unit_iter() (translate.storage.trados.TradosUnit method), 678 unit_iter() (translate.storage.ts2.tsfile method), 683 (translate.storage.ts2.tsunit method), unit iter() 686 unit_iter() (translate.storage.txt.TxtFile method), 688 unit_iter() (translate.storage.txt.TxtUnit method), 691 unit_iter() (translate.storage.utx.UtxFile method), unit_iter() (translate.storage.utx.UtxUnit method), 697 UnitClass 563 unit_iter() (translate.storage.wordfast.WordfastTMFile method), UnitClass unit iter() (trans-UnitClass late.storage.wordfast.WordfastUnit method), UnitClass unit_iter() (translate.storage.xliff.xlifffile method), UnitClass unit_iter() (translate.storage.xliff.xliffunit method), UnitClass unit_iter() (translate.storage.zip.ZIPFile method), 714 UnitClass UnitChecker (class in translate.filters.checks), 350 UnitClass (translate.storage.base.DictStore attribute), 403 UnitClass (translate.storage.base.TranslationStore attribute), 408 UnitClass (translate.storage.catkeys.CatkeysFile at-

tribute), 415

421

UnitClass (translate.storage.csvl10n.csvfile attribute),

UnitClass (translate.storage.dtd.dtdfile attribute), 427 UnitClass (translate.storage.html.htmlfile attribute), UnitClass (translate.storage.html.POHTMLParser at*tribute*), 433 UnitClass (translate.storage.ical.icalfile attribute), UnitClass (translate.storage.ini.inifile attribute), 446 (translate.storage.jsonl10n.ARBJsonFile attribute), 452 UnitClass (translate.storage.jsonl10n.GoI18NJsonFile attribute), 457 UnitClass (translate.storage.jsonl10n.I18NextFile attribute), 462 UnitClass (translate.storage.jsonl10n.JsonFile attribute), 467 UnitClass (translate.storage.jsonl10n.JsonNestedFile attribute), 469 UnitClass (translate.storage.jsonl10n.WebExtensionJsonFile attribute), 477 UnitClass (translate.storage.lisa.LISAfile attribute), UnitClass (translate.storage.mo.mofile attribute), 487 UnitClass (translate.storage.mozilla lang.LangStore attribute), 493 UnitClass (translate.storage.omegat.OmegaTFile attribute), 498 UnitClass (translate.storage.omegat.OmegaTFileTab attribute), 500 UnitClass (translate.storage.oo.oofile attribute), 506 UnitClass (translate.storage.php.LaravelPHPFile at*tribute*), 558 (translate.storage.php.phpfile attribute), (translate.storage.pocommon.pofile *tribute*), 568 (translate.storage.poxliff.PoXliffFile tribute), 576 (translate.storage.properties.gwtfile tribute), 595 (translate.storage.properties.javafile attribute), 598 (translate.storage.properties.javautf16file attribute), 599 (translate.storage.properties.javautf8file attribute), 601 UnitClass (translate.storage.properties.joomlafile at*tribute*), 603 UnitClass (translate.storage.properties.propfile attribute), 605 UnitClass (translate.storage.properties.stringsfile at-

Index 867

tribute), 613

attribute), 614

UnitClass (translate.storage.properties.stringsutf8file

UnitClass (translate.storage.properties.xwikifile attribute), 616	untranslated() (trans- late.filters.checks.DrupalChecker method),
UnitClass (translate.storage.properties.XWikiFullPage	284
attribute), 592	untranslated() (trans-
UnitClass (translate.storage.properties.XWikiPagePropattribute), 593	erties late.filters.checks.GnomeChecker method), 290
	untranslated() (translate.filters.checks.IOSChecker method), 295
UnitClass (translate.storage.qm.qmfile attribute), 629	untranslated() (trans-
UnitClass (translate.storage.qph.QphFile attribute), 634	late.filters.checks.KdeChecker method), 301
UnitClass (translate.storage.rc.rcfile attribute), 640	untranslated() (trans-
UnitClass (translate.storage.subtitles.AdvSubStationAlpattribute), 647	phaFile late.filters.checks.L20nChecker method), 307
${\tt UnitClass} \ \ ({\it translate.storage.subtitles.MicroDVDFile}$	untranslated() (trans-
attribute), 648	$late. filters. checks. Libre Of fice Checker\ method),$
UnitClass (translate.storage.subtitles.SubRipFile at-	313
tribute), 650	untranslated() (trans-
UnitClass (translate.storage.subtitles.SubStationAlphaFattribute), 652	318
UnitClass (translate.storage.subtitles.SubtitleFile at-	untranslated() (trans-
tribute), 654	late.filters.checks.MozillaChecker method),
UnitClass (translate.storage.tbx.tbxfile attribute), 659	324
UnitClass (translate.storage.tiki.TikiStore attribute),	untranslated() (trans-
664	late.filters.checks.OpenOfficeChecker method), 330
UnitClass (translate.storage.tmx.tmxfile attribute), 669	untranslated() (trans-
UnitClass (translate.storage.trados.TradosTxtTmFile	late.filters.checks.ReducedChecker method),
attribute), 679	336
UnitClass (translate.storage.ts2.tsfile attribute), 681	untranslated() (trans-
UnitClass (translate.storage.txt.TxtFile attribute), 686	late.filters.checks.StandardChecker method),
UnitClass (translate.storage.utx.UtxFile attribute),	341
692	untranslated() (trans-
UnitClass (translate.storage.wordfast.WordfastTMFile attribute), 698	late.filters.checks.TermChecker method), 349
UnitClass (translate.storage.xliff.xlifffile attribute),	untranslated_unitcount() (trans-
704	late.storage.statistics.Statistics method),
UnitMixer (class in translate.convert.accesskey), 236	645
unitstats() (translate.storage.statsdb.StatsCache	untranslated_units() (trans-
method), 646	late.storage.statistics.Statistics method),
UnknownExtensionError, 246	645
UnknownXML (class in trans-	untranslated_wordcount() (trans-
late.storage.placeables.xliff), 556	late.storage.statistics.Statistics method),
unormalizechar (class in translate.storage.oo), 507	645
unquote_plus() (in module trans-	update() (in module translate.storage.poheader), 575
late.storage.pocommon), 573	update() (translate.filters.checks.CheckerConfig
unquotefromandroid() (in module trans-	method), 279
late.storage.dtd), 432	update() (translate.misc.dictutils.cidict method), 389
unquotefromdtd() (in module trans-	update() (translate.storage.oo.unormalizechar
late.storage.dtd), 432	method), 508
UnsupportedConversionError, 246	update_file() (trans-
untranslated() (trans-	late.storage.bundleprojstore.BundleProjectStore
late.filters.checks.CCLicenseChecker method),	method), 415
278	<pre>update_file() (translate.storage.project.Project</pre>

method), 584	urls() (translate.filters.checks.KdeChecker method),
update_file() (trans-	301
late.storage.projstore.ProjectStore method), 585	urls() (translate.filters.checks.L20nChecker method), 307
updatecontributor() (translate.storage.mo.mofile method), 490	urls() (translate.filters.checks.LibreOfficeChecker method), 313
updatecontributor() (trans-	urls() (translate.filters.checks.MinimalChecker
late.storage.pocommon.pofile method), 570	method), 319
updatecontributor() (trans-	urls() (translate.filters.checks.MozillaChecker
late.storage.poheader.poheader method),	method), 324
575	urls() (translate.filters.checks.OpenOfficeChecker
updatecontributor() (trans-	method), 330
late.storage.poxliff.PoXliffFile method), 579	urls() (translate.filters.checks.ReducedChecker
updatecontributor() (trans-	method), 336
late.storage.pypo.pofile method), 624	urls() (translate.filters.checks.StandardChecker
updateheader() (translate.storage.mo.mofile	method), 342
method), 490	urls() (translate.filters.checks.TermChecker method),
	349
updateheader() (translate.storage.pocommon.pofile method), 570	usable() (translate.search.match.matcher method),
updateheader() (trans-	402
late.storage.poheader.poheader method), 575	usable() (translate.search.match.terminologymatcher method), 403
updateheader() (translate.storage.poxliff.PoXliffFile method), 579	UtxDialect (class in translate.storage.utx), 692 UtxFile (class in translate.storage.utx), 692
updateheader() (translate.storage.pypo.pofile method), 624	UtxHeader (class in translate.storage.utx), 693 UtxUnit (class in translate.storage.utx), 694
updateheaderplural() (trans-	,,,,
late.storage.mo.mofile method), 490	V
updateheaderplural() (trans-	<pre>valid_fieldnames() (in module trans-</pre>
()	valia_lielanames() (in mounte trans-
late.storage.pocommon.pofile method), 570	late storage csv110n) 426
late.storage.pocommon.pofile method), 570	late.storage.csvl10n), 426
updateheaderplural() (trans-	validaccel (translate.lang.common.Common at-
updateheaderplural() (trans- late.storage.poheader.poheader method),	validaccel (translate.lang.common.Common attribute), 364
updateheaderplural() (trans- late.storage.poheader.poheader method), 575	<pre>validaccel (translate.lang.common.Common at-</pre>
<pre>updateheaderplural() (trans- late.storage.poheader.poheader</pre>	validaccel (translate.lang.common.Common at- tribute), 364 validchars() (trans- late.filters.checks.CCLicenseChecker method),
updateheaderplural() (trans- late.storage.poheader.poheader method), 575 updateheaderplural() (trans- late.storage.poxliff.PoXliffFile method), 579	validaccel (translate.lang.common.Common at- tribute), 364 validchars() (trans- late.filters.checks.CCLicenseChecker method), 278
updateheaderplural() (trans- late.storage.poheader.poheader method), 575 updateheaderplural() (trans- late.storage.poxliff.PoXliffFile method), 579 updateheaderplural() (trans-	<pre>validaccel (translate.lang.common.Common at-</pre>
updateheaderplural() (trans- late.storage.poheader.poheader method), 575 updateheaderplural() (trans- late.storage.poxliff.PoXliffFile method), 579 updateheaderplural() (trans- late.storage.pypo.pofile method), 624	<pre>validaccel (translate.lang.common.Common at-</pre>
updateheaderplural() (trans- late.storage.poheader.poheader method), 575 updateheaderplural() (trans- late.storage.poxliff.PoXliffFile method), 579 updateheaderplural() (trans- late.storage.pypo.pofile method), 624 updatetargetlanguage() (trans-	validaccel (translate.lang.common.Common attribute), 364 validchars() (translate.filters.checks.CCLicenseChecker method), 278 validchars() (translate.filters.checks.DrupalChecker method), 284
updateheaderplural() (trans- late.storage.poheader.poheader method), 575 updateheaderplural() (trans- late.storage.poxliff.PoXliffFile method), 579 updateheaderplural() (trans- late.storage.pypo.pofile method), 624 updatetargetlanguage() (trans- late.filters.checks.CheckerConfig method),	<pre>validaccel (translate.lang.common.Common at-</pre>
updateheaderplural() (trans- late.storage.poheader.poheader method), 575 updateheaderplural() (trans- late.storage.poxliff.PoXliffFile method), 579 updateheaderplural() (trans- late.storage.pypo.pofile method), 624 updatetargetlanguage() (trans- late.filters.checks.CheckerConfig method), 279	<pre>validaccel (translate.lang.common.Common at-</pre>
updateheaderplural() (trans- late.storage.poheader.poheader method), 575 updateheaderplural() (trans- late.storage.poxliff.PoXliffFile method), 579 updateheaderplural() (trans- late.storage.pypo.pofile method), 624 updatetargetlanguage() (trans- late.filters.checks.CheckerConfig method), 279 updatevalidchars() (trans-	validaccel (translate.lang.common.Common attribute), 364 validchars() (translate.filters.checks.CCLicenseChecker method), 278 validchars() (translate.filters.checks.DrupalChecker method), 284 validchars() (translate.filters.checks.GnomeChecker method), 290
updateheaderplural() (trans- late.storage.poheader.poheader method), 575 updateheaderplural() (trans- late.storage.poxliff.PoXliffFile method), 579 updateheaderplural() (trans- late.storage.pypo.pofile method), 624 updatetargetlanguage() (trans- late.filters.checks.CheckerConfig method), 279 updatevalidchars() (trans- late.filters.checks.CheckerConfig method), 279	<pre>validaccel (translate.lang.common.Common at-</pre>
updateheaderplural() (trans- late.storage.poheader.poheader poheaderplural() (trans- late.storage.poxliff.PoXliffFile method), 579 updateheaderplural() (trans- late.storage.pypo.pofile method), 624 updatetargetlanguage() (trans- late.filters.checks.CheckerConfig pupdatevalidchars() (trans- late.filters.checks.CheckerConfig	validaccel (translate.lang.common.Common attribute), 364 validchars() (translate.filters.checks.CCLicenseChecker method), 278 validchars() (translate.filters.checks.DrupalChecker method), 284 validchars() (translate.filters.checks.IOSChecker method), 290 validchars() (translate.filters.checks.IOSChecker method), 296 validchars() (translate.filters.checks.KdeChecker
updateheaderplural() (trans- late.storage.poheader.poheader method), 575 updateheaderplural() (trans- late.storage.poxliff.PoXliffFile method), 579 updateheaderplural() (trans- late.storage.pypo.pofile method), 624 updatetargetlanguage() (trans- late.filters.checks.CheckerConfig method), 279 updatevalidchars() (trans- late.filters.checks.CheckerConfig method), 279 upper() (translate.misc.multistring.multistring	validaccel (translate.lang.common.Common attribute), 364 validchars () (translate.filters.checks.CCLicenseChecker method), 278 validchars () (translate.filters.checks.DrupalChecker method), 284 validchars () (translate.filters.checks.IOSChecker method), 290 validchars () (translate.filters.checks.IOSChecker method), 296 validchars () (translate.filters.checks.KdeChecker method), 301
updateheaderplural() (trans- late.storage.poheader.poheader method), 575 updateheaderplural() (trans- late.storage.poxliff.PoXliffFile method), 579 updateheaderplural() (trans- late.storage.pypo.pofile method), 624 updatetargetlanguage() (trans- late.filters.checks.CheckerConfig method), 279 updatevalidchars() (trans- late.filters.checks.CheckerConfig method), 279 upper() (translate.misc.multistring.multistring method), 393 ur (class in translate.lang.ur), 386 urls() (translate.filters.checks.CCLicenseChecker	validaccel (translate.lang.common.Common attribute), 364 validchars() (translate.filters.checks.CCLicenseChecker method), 278 validchars() (translate.filters.checks.DrupalChecker method), 284 validchars() (translate.filters.checks.IOSChecker method), 290 validchars() (translate.filters.checks.IOSChecker method), 296 validchars() (translate.filters.checks.KdeChecker
updateheaderplural() (trans- late.storage.poheader.poheader	validaccel (translate.lang.common.Common attribute), 364 validchars() (translate.filters.checks.CCLicenseChecker method), 278 validchars() (translate.filters.checks.DrupalChecker method), 284 validchars() (translate.filters.checks.IOSChecker method), 290 validchars() (translate.filters.checks.IOSChecker method), 296 validchars() (translate.filters.checks.KdeChecker method), 301 validchars() (translate.filters.checks.L20nChecker
updateheaderplural() (trans- late.storage.poheader.poheader method), 575 updateheaderplural() (trans- late.storage.poxliff.PoXliffFile method), 579 updateheaderplural() (trans- late.storage.pypo.pofile method), 624 updatetargetlanguage() (trans- late.filters.checks.CheckerConfig method), 279 updatevalidchars() (trans- late.filters.checks.CheckerConfig method), 279 upper() (translate.misc.multistring.multistring method), 393 ur (class in translate.lang.ur), 386 urls() (translate.filters.checks.CCLicenseChecker	validaccel (translate.lang.common.Common attribute), 364 validchars() (translate.filters.checks.CCLicenseChecker method), 278 validchars() (translate.filters.checks.DrupalChecker method), 284 validchars() (translate.filters.checks.GnomeChecker method), 290 validchars() (translate.filters.checks.IOSChecker method), 296 validchars() (translate.filters.checks.KdeChecker method), 301 validchars() (translate.filters.checks.L20nChecker method), 307
updateheaderplural() (trans- late.storage.poheader.poheader method), 575 updateheaderplural() (trans- late.storage.poxliff.PoXliffFile method), 579 updateheaderplural() (trans- late.storage.pypo.pofile method), 624 updatetargetlanguage() (trans- late.filters.checks.CheckerConfig method), 279 updatevalidchars() (trans- late.filters.checks.CheckerConfig method), 279 upper() (translate.misc.multistring.multistring method), 393 ur (class in translate.lang.ur), 386 urls() (translate.filters.checks.CCLicenseChecker method), 278 urls() (translate.filters.checks.DrupalChecker	validaccel (translate.lang.common.Common attribute), 364 validchars() (translate.filters.checks.CCLicenseChecker method), 278 validchars() (translate.filters.checks.DrupalChecker method), 284 validchars() (translate.filters.checks.GnomeChecker method), 290 validchars() (translate.filters.checks.IOSChecker method), 296 validchars() (translate.filters.checks.KdeChecker method), 301 validchars() (translate.filters.checks.L20nChecker method), 307 validchars() (translate.filters.checks.L20nChecker method), 307 validchars() (translate.filters.checks.L20nChecker method), 307
updateheaderplural() (trans- late.storage.poheader.poheader method), 575 updateheaderplural() (trans- late.storage.poxliff.PoXliffFile method), 579 updateheaderplural() (trans- late.storage.pypo.pofile method), 624 updatetargetlanguage() (trans- late.filters.checks.CheckerConfig method), 279 updatevalidchars() (trans- late.filters.checks.CheckerConfig method), 279 upper() (translate.misc.multistring.multistring method), 393 ur (class in translate.lang.ur), 386 urls() (translate.filters.checks.CCLicenseChecker method), 278 urls() (translate.filters.checks.DrupalChecker method), 284	validaccel (translate.lang.common.Common attribute), 364 validchars () (translate.filters.checks.CCLicenseChecker method), 278 validchars () (translate.filters.checks.DrupalChecker method), 284 validchars () (translate.filters.checks.IOSChecker method), 290 validchars () (translate.filters.checks.IOSChecker method), 296 validchars () (translate.filters.checks.KdeChecker method), 301 validchars () (translate.filters.checks.L20nChecker method), 307 validchars () (translate.filters.checks.L20nChecker method), 307 validchars () (translate.filters.checks.L20nChecker method), 313

validchars() late.filters.checks.MozillaChecker	(trans- method),	late.storage.properties.DialectXWiki class method), 592
325	memou),	values () (translate.misc.dictutils.cidict method), 389
validchars()	(trans-	values () (translate.state.tuttis.ctatet method), 369 values () (translate.storage.oo.unormalizechar
late.filters.checks.OpenOfficeChecker		method), 508
330	memou),	variables() (trans-
validchars()	(trans-	late.filters.checks.CCLicenseChecker method),
		278
late.filters.checks.ReducedChecker 336	method),	variables() (translate.filters.checks.DrupalChecker
validchars()	(trans-	method), 284
late.filters.checks.StandardChecker 342	method),	variables () (translate.filters.checks.GnomeChecker method), 290
validchars () (translate.filters.checks.Termethod), 349	mChecker	variables () (translate.filters.checks.IOSChecker method), 296
validdoublewords	(trans-	variables() (translate.filters.checks.KdeChecker
late.lang.common.Common attribute)		method), 302
validxml() (translate.filters.checks.LibreOff method), 313		
value_strip() (translate.storage.properti	es.Dialect	
class method), 586		late.filters.checks.LibreOfficeChecker method),
<pre>value_strip()</pre>	(trans-	313
late.storage.properties.DialectFlex	class	variables() (trans-
method), 587		late.filters.checks.MinimalChecker method),
<pre>value_strip()</pre>	(trans-	319
late.storage.properties.DialectGaia	class	variables () (translate.filters.checks.MozillaChecker
method), 587		method), 325
value_strip()	(trans-	variables() (trans-
late.storage.properties.DialectGwt method), 588	class	late.filters.checks.OpenOfficeChecker method), 330
value_strip()	(trans-	variables() (trans-
late.storage.properties.DialectJava	class	late.filters.checks.ReducedChecker method),
method), 588		336
<pre>value_strip()</pre>		variables() (trans-
late.storage.properties.DialectJavaUt class method), 589	f16	late.filters.checks.StandardChecker method), 342
<pre>value_strip()</pre>	(trans-	variables() (translate.filters.checks.TermChecker
late.storage.properties.DialectJavaUṭ	f8 class	method), 349
method), 589		varname() (in module translate.filters.prefilters), 356
<pre>value_strip()</pre>	(trans-	varnone () (in module translate.filters.prefilters), 356
late.storage.properties.DialectJoomla method), 589	class	VerboseProgressBar (class in trans- late.misc.progressbar), 398
value_strip()	(trans-	verifyoptions() (in module trans-
late.storage.properties.DialectMozilla	*	late.convert.oo2po), 249
method), 590	e crass	verifyoptions() (in module trans-
value_strip()	(trans-	late.convert.oo2xliff), 249
late.storage.properties.DialectSkype	class	verifyoptions() (trans-
method), 590		late.convert.convert.ArchiveConvertOptionParser
value_strip()	(trans-	method), 241
late.storage.properties.DialectStrings		verifyoptions() (trans-
method), 591		late.convert.convert.ConvertOptionParser
value_strip()	(trans-	method), 244
late.storage.properties.DialectStrings	`	verifyoptions() (trans-
class method), 591	*	late.convert.po2moz.MozConvertOptionParser
value_strip()	(trans-	method), 256

verifyoptions() late.convert.po2tmx.TmxOptionParser method), 261	(trans-	with_traceback() late.convert.prop2po.DiscardUnit 268	(trans- method),
verifyoptions() late.convert.po2wordfast.WfOptionPars. method), 266	(trans- er	with_traceback() late.filters.checks.FilterFailure 285	(trans- method),
vi (class in translate.lang.vi), 387		with_traceback()	(trans-
W		late.filters.checks.SeriousFilterFailure method), 336	?
warning()(translate.convert.convert.ArchiveConvert.		late.storage.base.ParseError method),	(<i>trans</i> - , 408
warning() (translate.convert.convert.ConvertO method), 244	ptionPar.	seyith_traceback() late.storage.bundleprojstore.InvalidBu	(trans- ındleError
warning()(translate.convert.po2moz.MozConv method), 256	vertOption	nParser method), 415 with_traceback()	(trans-
warning() (translate.convert.po2tmx.TmxOption method), 261	onParser	late.storage.placeables.strelem.Eleme.method), 537	ntNotFoundErro
warning() (translate.convert.po2wordfast.WfO method), 266		serith_traceback() late.storage.projstore.FileExistsInProj method), 584	(trans- iectError
warning() (translate.filters.pofilter.FilterOption method), 355	nParser	with_traceback()	(trans-
warning() (translate.misc.optrecurse.Recursive method), 396	eOptionP		
warning() (translate.tools.poconflicts.Conflict	OptionPa	rwith_traceback()	(trans-
method), 718		late.storage.tmdb.LanguageError	method),
warning() (translate.tools.pogrep.GrepOptio	nParser	669	(4
<pre>method), 722 warning() (translate.tools.porestructure.SplitO method), 725</pre>)ptionPar	<pre>with_traceback() ser late.storage.workflow.InvalidStateObj method), 704</pre>	(trans- ectError
warning() (translate.tools.poterminology.Term method), 729	inologyO	pฟีเอ็กParระลิaceback () late.storage.workflow.NoInitialStateEi	(trans- rror
WebExtensionJsonFile (class in	trans-	method), 704	
late.storage.jsonl10n), 477		<pre>with_traceback()</pre>	(trans-
WebExtensionJsonUnit (class in late.storage.jsonl10n), 478	trans-	late.storage.workflow.StateNotInWork method), 704	flowError
WF_ESCAPE_MAP (in module	trans-	<pre>with_traceback() late.storage.workflow.TransitionError</pre>	(trans-
late.storage.wordfast), 698 WF_FIELDNAMES (in module	trans	704	memoa),
late.storage.wordfast), 698	trans-	with_traceback()	(trans-
WF_FIELDNAMES_HEADER (in module late.storage.wordfast), 698	trans-	late.storage.workflow.WorkflowError	method),
WF_FIELDNAMES_HEADER_DEFAULTS (in translate.storage.wordfast), 698	module	<pre>word_iter() (translate.lang.af.af class meth word_iter() (translate.lang.am.am class</pre>	
WF_TIMEFORMAT (in module	trans-	358	N 250
late.storage.wordfast), 698		<pre>word_iter() (translate.lang.ar.ar class methods</pre> word_iter() (translate.lang.bn.bn class methods	
WfOptionParser (class in late.convert.po2wordfast), 263	trans-	<pre>word_iter() (translate.lang.code_or.code_</pre>	
with_traceback()	(trans- Error	method), 360 word_iter() (translate.lang.common.Comm	non class
method), 246	(,	method), 364	had) 366
with_traceback()	(trans-	word_iter() ($translate.lang.de.de$ $class$ $meth_r$ word_iter() ($translate.lang.el.el$ $class$ $meth$	
method), 246	SIONETTO	word_iter() (translate.lang.es.es class method word_iter() (translate.lang.es.es class method word_iter() (translate.lang.fa.fa class method)	nod), 367

word_iter() (translate.lang.fi.fi class method), 369	words () (translate.lang.hy.hy class method), 373	
word_iter() (translate.lang.fr.fr class method), 370	words () (translate.lang.ja.ja class method), 374	
<pre>word_iter() (translate.lang.gu.gu class method), 371</pre>	words () (translate.lang.km.km class method), 375	
<pre>word_iter() (translate.lang.he.he class method), 372</pre>	words () (translate.lang.kn.kn class method), 376	
word_iter() (translate.lang.hi.hi class method), 372	words () (translate.lang.ko.ko class method), 376	
<pre>word_iter() (translate.lang.hy.hy class method), 373</pre>	words () (translate.lang.ml.ml class method), 377	
<pre>word_iter() (translate.lang.ja.ja class method), 374</pre>	words () (translate.lang.mr.mr class method), 378	
<pre>word_iter() (translate.lang.km.km class method),</pre>	words () (translate.lang.ne.ne class method), 379	
375	words () (translate.lang.pa.pa class method), 380	
<pre>word_iter() (translate.lang.kn.kn class method), 376</pre>	words () (translate.lang.si.si class method), 381	
word_iter() (translate.lang.ko.ko class method), 376	words () (translate.lang.st.st class method), 382	
word_iter() (translate.lang.ml.ml class method), 377	words () (translate.lang.sv.sv class method), 383	
word_iter() (translate.lang.mr.mr class method), 378	words () (translate.lang.ta.ta class method), 383	
word_iter() (translate.lang.ne.ne class method), 379	words () (translate.lang.te.te class method), 384	
word_iter() (translate.lang.pa.pa class method), 380	words () (translate.lang.th.th class method), 385	
word_iter() (translate.lang.si.si class method), 381	words () (translate.lang.ug.ug class method), 386	
word_iter() (translate.lang.st.st class method), 382	words () (translate.lang.ur.ur class method), 387	
word_iter() (translate.lang.sv.sv class method), 383	words () (translate.lang.vi.vi class method), 387	
word_iter() (translate.lang.ta.ta class method), 383	words () (translate.lang.zh.zh class method), 388	
word_iter() (translate.lang.te.te class method), 384	wordsinunit() (in module translate.storage.statsdb),	
word_iter() (translate.lang.th.th class method), 385	646	
word_iter() (translate.lang.ug.ug class method), 386	WorkflowError, 704	
word_iter() (translate.lang.ur.ur class method), 387	wrap() (translate.storage.pypo.PoWrapper method),	
word_iter() (translate.lang.vi.vi class method), 387	621	
word_iter() (translate.lang.zh.zh class method), 388	<pre>wrap_production() (in module trans-</pre>	
wordcount() (translate.storage.statistics.Statistics	late.storage.php), 568	
method), 645	wrapmessage() (translate.convert.po2txt.po2txt	
WordfastDialect (class in trans-	method), 262	
late.storage.wordfast), 698	<pre>write_odf() (in module translate.convert.xliff2odf),</pre>	
WordfastHeader (class in trans-	271	
late.storage.wordfast), 698	<pre>writediff() (translate.tools.pydiff.DirDiffer</pre>	
WordfastTime (class in translate.storage.wordfast),	method), 730	
700	<pre>writediff() (translate.tools.pydiff.FileDiffer</pre>	
WordfastTMFile (class in trans-	method), 730	
late.storage.wordfast), 698	<pre>writexml_helper() (in module trans-</pre>	
WordfastUnit (class in translate.storage.wordfast),	late.misc.ourdom), 398	
700	,,,	
words () (translate.lang.af.af class method), 357	X	
words () (translate.lang.am.am class method), 358		
words () (translate.lang.ar.ar class method), 359	X (class in translate.storage.placeables.base), 519	
words () (translate.lang.bn.bn class method), 360	X (class in translate.storage.placeables.xliff), 544	
words () (translate.lang.code_or.code_or class	xlifffile (class in translate.storage.xliff), 704	
	xliffunit (class in translate.storage.xliff), 707	
method), 360	xml_preserve_ancestors (in module trans-	
words() (translate.lang.common.Common class method), 364	late mise vml helpers) 400	
MP111001 304	late.misc.xml_helpers), 400	
	xml_space_ancestors (in module trans-	
words () (translate.lang.de.de class method), 366	<pre>xml_space_ancestors (in module trans- late.misc.xml_helpers), 401</pre>	
words () (translate.lang.de.de class method), 366 words () (translate.lang.el.el class method), 367	<pre>xml_space_ancestors (in module trans- late.misc.xml_helpers), 401 XMLEntityPlaceable (class in trans-</pre>	
words () (translate.lang.de.de class method), 366 words () (translate.lang.el.el class method), 367 words () (translate.lang.es.es class method), 368	<pre>xml_space_ancestors (in module trans- late.misc.xml_helpers), 401 XMLEntityPlaceable (class in trans- late.storage.placeables.general), 524</pre>	
words () (translate.lang.de.de class method), 366 words () (translate.lang.el.el class method), 367 words () (translate.lang.es.es class method), 368 words () (translate.lang.fa.fa class method), 369	<pre>xml_space_ancestors (in module trans- late.misc.xml_helpers), 401 XMLEntityPlaceable (class in trans- late.storage.placeables.general), 524 XmlNamer (class in translate.storage.xml_name), 713</pre>	
words () (translate.lang.de.de class method), 366 words () (translate.lang.el.el class method), 367 words () (translate.lang.es.es class method), 368 words () (translate.lang.fa.fa class method), 369 words () (translate.lang.fi.fi class method), 369	<pre>xml_space_ancestors (in module trans- late.misc.xml_helpers), 401 XMLEntityPlaceable (class in trans- late.storage.placeables.general), 524 XmlNamer (class in translate.storage.xml_name), 713 XMLTagPlaceable (class in trans-</pre>	
words () (translate.lang.de.de class method), 366 words () (translate.lang.el.el class method), 367 words () (translate.lang.es.es class method), 368 words () (translate.lang.fa.fa class method), 369 words () (translate.lang.fi.fi class method), 369 words () (translate.lang.fr.fr class method), 370	<pre>xml_space_ancestors (in module trans- late.misc.xml_helpers), 401 XMLEntityPlaceable (class in trans- late.storage.placeables.general), 524 XmlNamer (class in translate.storage.xml_name), 713</pre>	
words () (translate.lang.de.de class method), 366 words () (translate.lang.el.el class method), 367 words () (translate.lang.es.es class method), 368 words () (translate.lang.fa.fa class method), 369 words () (translate.lang.fi.fi class method), 369 words () (translate.lang.fr.fr class method), 370 words () (translate.lang.gu.gu class method), 371	<pre>xml_space_ancestors (in module trans- late.misc.xml_helpers), 401 XMLEntityPlaceable (class in trans- late.storage.placeables.general), 524 XmlNamer (class in translate.storage.xml_name), 713 XMLTagPlaceable (class in trans-</pre>	
words () (translate.lang.de.de class method), 366 words () (translate.lang.el.el class method), 367 words () (translate.lang.es.es class method), 368 words () (translate.lang.fa.fa class method), 369 words () (translate.lang.fi.fi class method), 369 words () (translate.lang.fr.fr class method), 370	<pre>xml_space_ancestors (in module trans- late.misc.xml_helpers), 401 XMLEntityPlaceable (class in trans- late.storage.placeables.general), 524 XmlNamer (class in translate.storage.xml_name), 713 XMLTagPlaceable (class in trans- late.storage.placeables.general), 526</pre>	

```
(translate.filters.checks.DrupalChecker
xmltags()
         method), 284
xmltags()
                 (translate.filters.checks.GnomeChecker
         method), 290
xmltags()
                    (translate.filters.checks.IOSChecker
         method), 296
                    (translate.filters.checks.KdeChecker
xmltags()
         method), 302
xmltags()
                  (translate.filters.checks.L20nChecker
         method), 308
xmltags() (translate.filters.checks.LibreOfficeChecker
         method), 313
               (translate.filters.checks.MinimalChecker
xmltags()
         method), 319
                (translate.filters.checks.MozillaChecker
xmltags()
         method), 325
xmltags()(translate.filters.checks.OpenOfficeChecker
         method), 331
               (translate.filters.checks.ReducedChecker
xmltags()
         method), 336
xmltags()
              (translate.filters.checks.StandardChecker
         method), 342
                   (translate.filters.checks.TermChecker
xmltags()
         method), 349
XPathBreadcrumb
                           (class
                                       in
                                                trans-
         late.storage.xml_extract.xpath_breadcrumb),
xwikifile (class in translate.storage.properties), 616
XWikiFullPage
                         (class
                                                trans-
         late.storage.properties), 592
XWikiPageProperties
                               (class
                                         in
                                                trans-
         late.storage.properties), 593
xwikiunit (class in translate.storage.properties), 618
yaml2po (class in translate.convert.yaml2po), 272
Ζ
zfill()
                  (translate.misc.multistring.multistring
         method), 393
zh (class in translate.lang.zh), 388
ZIPFile (class in translate.storage.zip), 714
```