Software Internationalization (i18n) with *GNU gettext*

Muhammad Najmi Ahmad Zabidi

Department of Computer Science
KICT, International Islamic University Malaysia,
MALAYSIA

25th October 2009

`najmi@kict.iiu.edu.my`

Created with LaTeX

Agenda

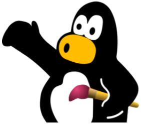
1 About
2 Internationalization
3 gettext
4 Task of developer/translator
5 Flow of the translation process
6 Codes
7 Tools
8 Route of PO to MO
9 Issues with package generated by gettext
10 Conclusion

About the speaker

- KDE Subversion committer and original translator to ms_MY

About the speaker

- KDE Subversion committer and original translator to ms_MY
- Translation CVS committer for Tuxpaint project



About the speaker

- KDE Subversion committer and original translator to ms_MY
- Translation CVS committer for Tuxpaint project
- Occasionally sending translation for GNU ms_MY, (Munsyi Project), lead by Sharuzzaman

About the speaker

- KDE Subversion committer and original translator to ms_MY
- Translation CVS committer for Tuxpaint project
- Occasionally sending translation for GNU ms_MY, (Munsyi Project), lead by Sharuzzaman

Agenda

1 About
2 Internationalization
3 gettext
4 Task of developer/translator
5 Flow of the translation process
6 Codes
7 Tools
8 Route of PO to MO
9 Issues with package generated by gettext
10 Conclusion

Internationalization

What actually internationalization is:

- Software going global

Internationalization

What actually internationalization is:

- Software going global
- Software package getting “world” acceptance, thus people willing to localize it

Internationalization

What actually internationalization is:

- Software going global
- Software package getting “world” acceptance, thus people willing to localize it
- Needs i18n support first!

The differences. . .

Localization

- Done by translator
- It can be by the user/Developer
- Submit to package maintainer
- Get credit

The differences. . .

Localization

- Done by translator
- It can be by the user/Developer
- Submit to package maintainer
- Get credit

Internationalization

- Developer
- Prepare package
- Merge package
- Update package
- Announce new package
- Give proper credit

Agenda

1. About
2. Internationalization
3. gettext
4. Task of developer/translator
5. Flow of the translation process
6. Codes
7. Tools
8. Route of PO to MO
9. Issues with package generated by gettext
10. Conclusion

gettext

gettext's features

- Part of GNU packages
- Enables internationalization of software
- Enables the creation of Portable Object (PO) file

Portable Object?

Gettext features

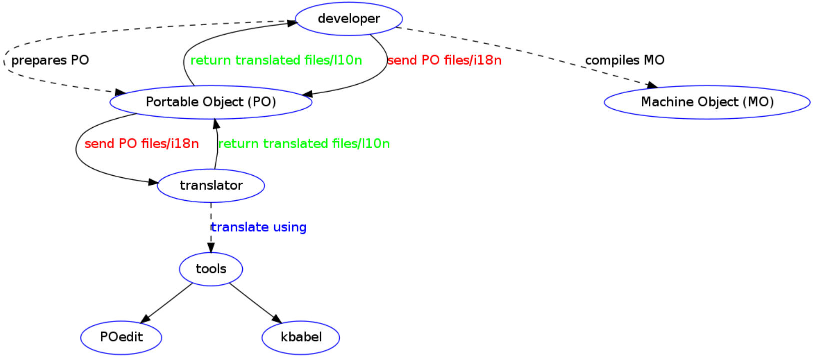
Why gettext?

- Supports major characters encoding . . .
- UTF-8 for an example
- KDE,Gnome, Squirrelmail use it
- Relatively to update and maintain

Agenda

1. About
2. Internationalization
3. gettext
4. Task of developer/translator
5. Flow of the translation process
6. Codes
7. Tools
8. Route of PO to MO
9. Issues with package generated by gettext
10. Conclusion

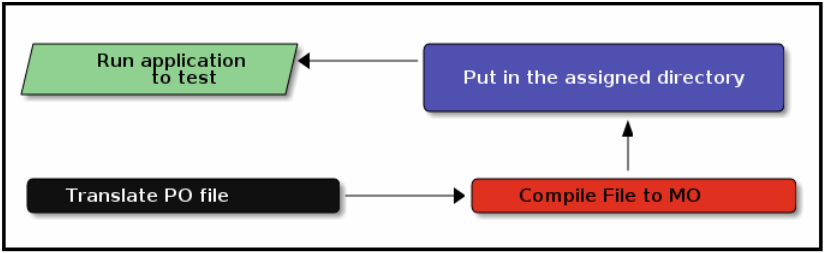
What developers and translators should do?



Agenda

1. About
2. Internationalization
3. gettext
4. Task of developer/translator
5. **Flow of the translation process**
6. Codes
7. Tools
8. Route of PO to MO
9. Issues with package generated by gettext
10. Conclusion

Flow of Translation



Portable Object (PO) files

PO file's features

- It's a raw, untranslated file
- Created automatically using gettext package
- Ready to be translated

Machine Object (MO) files

MO file's features

- Compiled file, derived from PO
- It's a binary, thus *machine readable*

Default & customized location

- By default Linux refers to

  `/usr/share/locale`

- Customized directory

  ```
  najmi@notre-dame:/var/www$ tree ms
  ms
  `-- LC_MESSAGES
      |-- greetings.mo
      |-- index.mo
      `-- messages.mo
  ```

Generating Machine Object file

How?

- Use msgfmt - message format
- This will generate default name, message.mo, unless you specify it
- You can trash it out, given you just want to check the localized stats to `/dev/null`
- It can be reverted back, use msgunfmt

Intro to locale

Locale

- Locale is the local setting of a particular country, race, venue etc
- For Bahasa Melayu, the assigned locale is ms_MY, where ms is for Malay, and MY is for Malaysia
- As for now, since Indonesia is using “id” , so sometimes ms is just fine
- Date sequence, for .eg dd/mm/yyyy is also fall under locale . We also do not have Daylight Saving Time(DST), that is also locale

Locale can be simply viewed by typing `$locale`

Some stuffs on chartset

Charset?

- Different charset support different language, characters
- Includes Roman, Arabic, CJK and etc
  - UTF-8, UTF-16 etc
- I don't really know on this stuffs, only know Roman based and Arabic based chars

Agenda

1. About
2. Internationalization
3. gettext
4. Task of developer/translator
5. Flow of the translation process
6. Codes
7. Tools
8. Route of PO to MO
9. Issues with package generated by gettext
10. Conclusion

How's the *C++* file looks like?

```
#include <iostream>
#include <libintl.h>
#include <locale>
using namespace std;

int main()
{
        setlocale( LC_ALL, "" );
        bindtextdomain( "hello", "/usr/share/locale" );
        textdomain( "hello" );
        cout<< gettext( "Hello,␣world!" ) <<endl;
        cout<< gettext( "How␣are␣you?" ) <<endl;
        return 0;
}
```

Executable output

```
$ ./hello
Hello, world!
How are you?
```

Executable output

```
$ ./hello
Hello, world!
How are you?
```

```
$ export LC_ALL=ms_MY.UTF-8
```

Executable output

```
$ ./hello
Hello, world!
How are you?
```

```
$ export LC_ALL=ms_MY.UTF-8

$ ./hello
Assalamualaikum, dunia!
Awak apa khabar?
```

About xgettext

- Part of gettext
- Extracting translatable strings from source code

```
xgettext -d lang lang.php
```

This will create lang.po, with respect to strings in lang.php

How's the *raw* POT file looks like?

```
# SOME DESCRIPTIVE TITLE.
# Copyright (C) YEAR THE PACKAGE'S COPYRIGHT HOLDER
# This file is distributed under the same license as the PACKAGE package.
# FIRST AUTHOR <EMAIL@ADDRESS>, YEAR.
#
#, fuzzy
msgid ""
msgstr ""
"Project-Id-Version: PACKAGE VERSION\n"
"Report-Msgid-Bugs-To: \n"
"POT-Creation-Date: 2009-10-23 22:08+0800\n"
"PO-Revision-Date: YEAR-MO-DA HO:MI+ZONE\n"
"Last-Translator: FULL NAME <EMAIL@ADDRESS>\n"
"Language-Team: LANGUAGE <LL@li.org>\n"
"MIME-Version: 1.0\n"
"Content-Type: text/plain; charset=CHARSET\n"
"Content-Transfer-Encoding: 8bit\n"

#: hello.cpp:11
msgid "Hello, world!"
msgstr ""

#: hello.cpp:12
msgid "How are you?"
msgstr ""
```

How’s the *translated* PO file looks like?

```
# translation of hello.po to Malay
# Muhammad Najmi bin Ahmad Zabidi <najmi.zabidi@gmail.com>, 2009.
msgid ""
msgstr ""
"Project-Id-Version: hello\n"
"Report-Msgid-Bugs-To: \n"
"POT-Creation-Date: 2009-10-23 22:08+0800\n"
"PO-Revision-Date: 2009-10-19 18:32+0800\n"
"Last-Translator: Muhammad Najmi bin Ahmad Zabidi <najmi.zabidi@gmail.com>\n"
"Language-Team: Malay <translation-team-ms@lists.sourceforge.net>\n"
"MIME-Version: 1.0\n"
"Content-Type: text/plain; charset=UTF-8\n"
"Content-Transfer-Encoding: 8bit\n"
"X-Generator: KBabel 1.11.4\n"

#: hello.cpp:11
msgid "Hello, world!"
msgstr "Assalamualaikum, dunia!"

#: hello.cpp:12
msgid "How are you?"
msgstr "Awak apa khabar?"
```

python!

```
import gettext
gettext.bindtextdomain('piton', '/usr/share/locale')
gettext.textdomain('piton')
_ = gettext.gettext
# ...
print _('python␣is␣simple')
print _('as␣simple␣as␣this')
```

Traslatable strings from Python file

```
# translation of piton.po to Malay
# Copyright (C) YEAR THE PACKAGE'S COPYRIGHT HOLDER
# This file is distributed under the same license as the PACKAGE package.
#
# Muhammad Najmi bin Ahmad Zabidi <najmi.zabidi@gmail.com>, 2009.
msgid ""
msgstr ""
"Project-Id-Version:␣piton\n"
"Report-Msgid-Bugs-To:␣\n"
"POT-Creation-Date:␣2009-10-24␣20:56+0800\n"
"PO-Revision-Date:␣2009-10-24␣20:57+0800\n"
"Last-Translator:␣Muhammad␣Najmi␣bin␣Ahmad␣Zabidi␣<najmi.zabidi@gmail.com>\n"
"Language-Team:␣Malay␣<kedidiemas@yahoogroups.com>\n"
"MIME-Version:␣1.0\n"
"Content-Type:␣text/plain;␣charset=UTF-8\n"
"Content-Transfer-Encoding:␣8bit\n"
"X-Generator:␣KBabel␣1.11.4\n"

#: piton.py:6
msgid "python␣is␣simple"
msgstr "python␣mudah"

#: piton.py:7
msgid "as␣simple␣as␣this"
msgstr "semudah␣ini"
```

Python executable

Python executable

```
$ python piton.py
python mudah
semudah ini
```

Python executable

```
$ python piton.py
python mudah
semudah ini
```

```
$ export LC_ALL=C
```

Python executable

```
$ python piton.py
python mudah
semudah ini
```

```
$ export LC_ALL=C
```

```
$ python piton.py
python is simple
as simple as this
```

Let’s see for PHP

```
<?
$language=ms_MY;
setlocale(LC_ALL,"$language");
bindtextdomain("index","/usr/share/locale/");
textdomain("index");
bind_textdomain_codeset("index", 'UTF–8');

echo _("Welcome\n");
echo "<br>";
echo _("Bye<br>");
echo gettext("The␣settings␣of␣").$language._("␣is␣working␣fine");
?>
```

Let’s see for PHP

- The following is the example of snipped file - without header

```
#: lang.php:8
msgid "Welcome\n"
msgstr ""

#: lang.php:10
msgid "Bye<br>"
msgstr ""

#: lang.php:11
msgid "The settings of "
msgstr ""

#: lang.php:11
msgid " is working fine"
msgstr ""
```

Avoid confusing translator

- Try to put strings in a minimal line
- Avoid separation of sentence

```
<?
$language=ms_MY;
setlocale(LC_ALL,"$language");
bindtextdomain("index","/usr/share/locale/");
textdomain("index");
bind_textdomain_codeset("index", 'UTF-8');

echo _("Welcome\n");
echo "<br>";
echo _("Bye<br>");
//echo _("The settings of ").$language._(" is working fine");
echo sprintf(_("The settings of %s is working fine"),$language);
?>
```

Use “sprintf”

Use the following . . .

```
echo sprintf(_(‘‘The settings of %s is working fine’’),$language);
```

instead of . . .

```
echo _(‘‘The settings of ’’).$language._(‘‘ is working fine”);
```

Separated into two

```
echo _(``The settings of '').$language._(`` is working fine");
```

will generate...

```
#: lang.php:11
msgid ``The settings of ''
msgstr ``

#: lang.php:11
msgid '' is working fine``
msgstr ''``''
```

which separates sentence into two different strings...

```
echo sprintf(_(``The settings of %s is working fine''),$language);
```

since this will generate. . .

```
msgid ``The settings of %s is working fine''
msgstr ``
```

What if we have plural noun. . .

- Plural is when you have different noun for singular and plural

Singular

- cat

Plural

- cats

Singular-Malay

Plural-Malay

What if we have plural noun. . .

- Plural is when you have different noun for singular and plural

Singular

- cat
- boy

Plural

- cats
- boys

Singular-Malay

Plural-Malay

What if we have plural noun. . .

- Plural is when you have different noun for singular and plural

Singular

- cat
- boy
- man

Plural

- cats
- boys
- men

Singular-Malay

Plural-Malay

What if we have plural noun. . .

- Plural is when you have different noun for singular and plural

Singular

- cat
- boy
- man

Plural

- cats
- boys
- men

Singular-Malay

- basikal

Plural-Malay

- basikal-basikal

What if we have plural noun. . .

- Plural is when you have different noun for singular and plural

Singular

- cat
- boy
- man

Plural

- cats
- boys
- men

Singular-Malay

- basikal
- pensil

Plural-Malay

- basikal-basikal
- pensil-pensil

What if we have plural noun. . .

- Plural is when you have different noun for singular and plural

Singular	Plural
cat	cats
boy	boys
man	men

Singular-Malay	Plural-Malay
basikal	basikal-basikal
pensil	pensil-pensil
komputer	komputer-komputer

However, as far as I know, plural issue in Malay languge is non trivial issue, so for the sake of easiness (perhaps, lazy?) plural = 0

ngettext

```php
<?php
$n = 2;
echo sprintf(ngettext("%d boy is eating", "%d boys are eating", $n), $n);
echo "<br>";
$n=1;
echo sprintf(ngettext("%d cat falls", "%d cats fall", $n), $n);
echo "<br>";
$n=1;
echo sprintf(ngettext("%d tingkap ditutup", "%d tetingkap ditutup", $n), $n);
echo "<br>";
$n=4;
//if it goes without parameter %d, no sprintf is fine
echo ngettext("File is good","Files are good",$n);
?>
```

```
2 boys are eating
1 cat falls
1 tingkap ditutup
Files are good
```

Sample PO

```
#: nplural.php:3
#, php-format
msgid "%d boy is eating"
msgid_plural "%d boys are eating"
msgstr[0] ""
msgstr[1] ""

#: nplural.php:6
#, php-format
msgid "%d cat falls"
msgid_plural "%d cats fall"
msgstr[0] ""
msgstr[1] ""

#: nplural.php:9
#, php-format
msgid "%d tingkap ditutup"
msgid_plural "%d tetingkap ditutup"
msgstr[0] ""
msgstr[1] ""

#: nplural.php:13
msgid "File is good"
msgid_plural "Files are good"
msgstr[0] ""
msgstr[1] ""
```

Agenda

1 About
2 Internationalization
3 gettext
4 Task of developer/translator
5 Flow of the translation process
6 Codes
7 Tools
8 Route of PO to MO
9 Issues with package generated by gettext
10 Conclusion

Internationalization tools

- gted (very recent)
- html2po / po2html

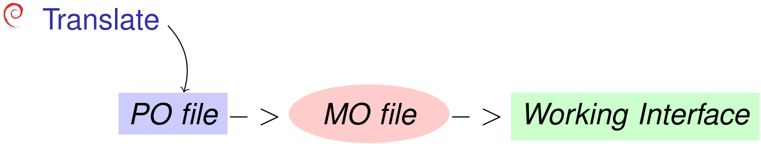
Localization tools

- POedit
- Kbabel (now lokalize)
- VI? Emacs ... the list may infinite up to *n* # of tools
- Incoming . . . wish for PO Live Edit, prototype is webl10n by gandalf
- Pootle

Agenda

1. About
2. Internationalization
3. gettext
4. Task of developer/translator
5. Flow of the translation process
6. Codes
7. Tools
8. Route of PO to MO
9. Issues with package generated by gettext
10. Conclusion

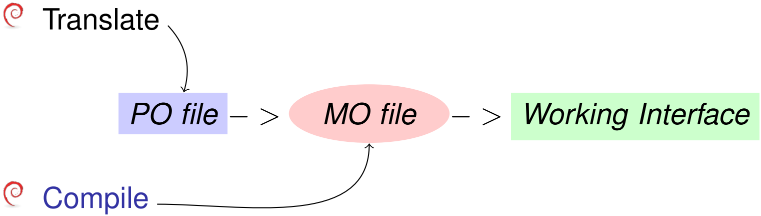
PO to MO journey



- Translate

PO file $->$ *MO file* $->$ *Working Interface*

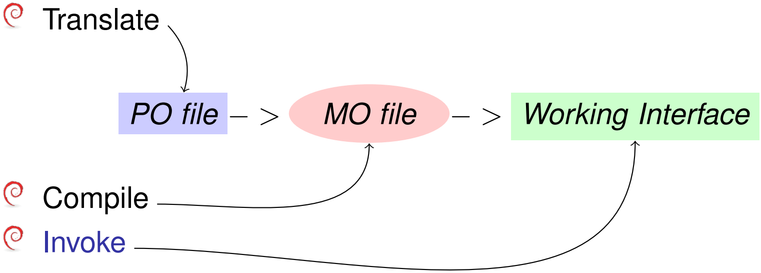
PO to MO journey



- Translate
- Compile

PO file – > *MO file* – > *Working Interface*

PO to MO journey



- Translate
- Compile
- Invoke

Agenda

1 About
2 Internationalization
3 gettext
4 Task of developer/translator
5 Flow of the translation process
6 Codes
7 Tools
8 Route of PO to MO
9 Issues with package generated by gettext
10 Conclusion

Issues

What are the issues?

- Software may become bloated

Issues

What are the issues?

- Software may become bloated
- So, the user may not install *everything* in the first place!

Issues

What are the issues?

- Software may become bloated
- So, the user may not install *everything* in the first place!
- Mozilla has its own way. . .
  - I saw moz2po and po2moz . . . a good news, perhaps?
  - Bug filed #501988 seems a good news

Issues

What are the issues?

- Software may become bloated
- So, the user may not install *everything* in the first place!
- Mozilla has its own way. . .
  - I saw moz2po and po2moz . . . a good news, perhaps?
  - Bug filed #501988 seems a good news
- I also have problem with UTF-8 on console.. garbled characters

Issues

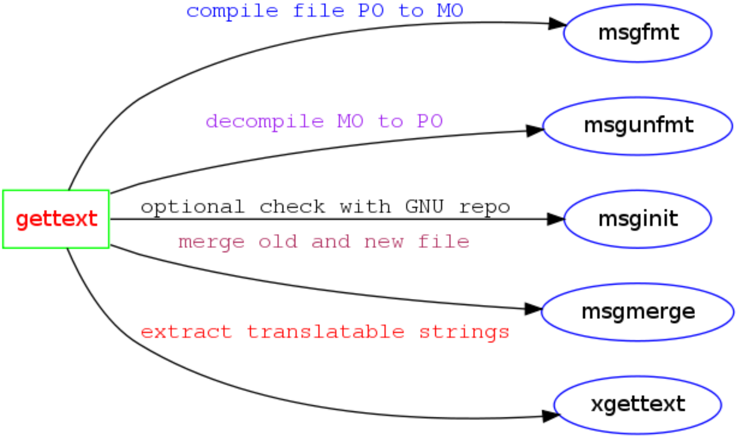
What are the issues?

- Software may become bloated
- So, the user may not install *everything* in the first place!
- Mozilla has its own way. . .
  - I saw moz2po and po2moz . . . a good news, perhaps?
  - Bug filed #501988 seems a good news
- I also have problem with UTF-8 on console.. garbled characters
  - Then, I `export LC_ALL=C`, of course I'm not happy with this
  - But most of the time, it works. . .

Agenda

1 About
2 Internationalization
3 gettext
4 Task of developer/translator
5 Flow of the translation process
6 Codes
7 Tools
8 Route of PO to MO
9 Issues with package generated by gettext
10 Conclusion

gettext rules



fin()

Questions?

هَل عِندَكُم السُّؤَال؟

najmi{at} kict.iiu.edu.my