Using translatexxe

Hussein Shafie, Pixware

December 2, 2005

Table of Contents

1. Introduction

`Translatexxe` is a command-line utility which can be used to translate messages (menu labels, button labels, error messages, etc) contained in XMLmind XML Editor (XXE for short).

Translatexxe is typically used by:

- an experienced XXE user,

- who is proficient in English,

- who considers that it is a shame XXE is not available in her/his mother tongue[1],

- who has a few hours to spare.

XMLmind will offer a ``lifetime'' Professional Edition license[2] to any person who will create and maintain an XXE translation to her/his mother tongue.

2. How does it work?

`Translatexxe` works in two steps:

1. Extract English messages from XXE code (in fact, from dozens of resource files contained XXE `.jar` files) and create a text file containing these messages.

   This text file contains placeholders for translated messages. The translator is supposed to type the text of translated messages in these placeholders using an ordinary text editor.

   This text file looks like this:

   ```
   ++++++++++com/xmlmind/xmledit/command/Messages.properties

   cancel=Cancel
   cancel=

   ### SelectById ---
   ```

---

[1]XMLmind will not hire professional translators to do the job. We want XMLmind XML Editor to be localized by users, that is by persons who understand not only the the meaning but also the use of the messages they translate.
[2]This person will be allowed to upgrade her/his XMLmind XML Editor Professional Edition as long as she/he wants to do so.

```
SBI.selectId=Select ID
SBI.selectId=

SBI.referenceToUndefinedId=Reference to non-existent ID "{0}".
SBI.referenceToUndefinedId=
        .
        .
        .
```

2. After the text file has been modified by the translator, convert this text file to an easy-to-deploy `.jar` file.

   If you want to test this newly created `.jar` file, simply copy this file to one of the two `addon/` directories and restart XXE.

   Similarly, if you want to share your work with other XXE users, simply put this `.jar` file on your web site and post an announcement to <xmleditor-support@xmlmind.com>. Then XXE users wishing to use your translation simply have to download the `.jar` file and copy it to their `addon/` directory.

The addon/ directory

The `addon/` directory is the place where XXE finds its extensions, *whatever is the type of this extension*: configurations, plug-ins, translations to languages other than the officially supported ones, dictionaries other than the bundled ones.

When XXE is started, it scans two `addon/` directories, searching them for extensions.

XXE_install_directory/addon/
      The system-wide `addon/` directory.

User_preferences_directory/addon/
      The personal `addon/` directory.

User_preferences_directory is:

- *$HOME*/.xxe/ on Linux and on the Mac (more generally, on Unix).

- *%SystemDrive%*\Documents and Settings*%USERNAME%*\Application Data\XMLmind\XMLeditor\ on Windows 2000/XP.

  If you cannot see the "Application Data" directory using Microsoft Windows File Manager, turn on Tools|Folder Options|View|File and Folders|Show hidden files and folders.

- *%SystemDrive%*\WINNT\Profiles*%USERNAME%*\Application Data\XMLmind\XMLeditor\ on Windows NT.

---

3. Procedure

As an example, we'll explain how to translate XXE messages to French[3].

3.1. First step: extract English messages

```
C:\Program Files\XMLmind_XML_Editor> translatexxe -m fr.txt
```

The above command creates a plain text file called "`fr.txt`" containing *all* XXE messages.

---

[3]This is just en example because XXE has already been translated to French.

Important

Unless this is the very first time you translate XXE messages, you need to execute a command other than "`translatexxe -m fr.txt`". More information in Section 3.5, "When a new version of XXE is released, update your translation".

By default, this text file is encoded using the native encoding of the platform (e.g. "`windows-1252`" for a West Europe PC running Windows) because it is assumed that the text editor you'll use to modify this file supports this encoding.

Tip

If you prefer, you can specify another encoding by using the "`-e`" option. Example:

```
C:\Program Files\XMLmind_XML_Editor> translatexxe \
    -e ISO-8859-1 -m fr.txt
```

If you don't know the exact name of an encoding, specify something like "`-e foo`" and `translatexxe` will report an error and will list all supported encodings.

"`Fr.txt`" looks like this:

```
++++++++++com/xmlmind/xmledit/command/Messages.properties❶

cancel❷=Cancel❸
cancel=❹

### SelectById ---❺

SBI.selectId=Select ID
SBI.selectId=

SBI.referenceToUndefinedId=Reference to non-existent ID "{0}".
SBI.referenceToUndefinedId=
    .
    .
    .
EM.repeatAction=Repeat {0}
EM.repeatAction=

----------com/xmlmind/xmledit/command/Messages.properties❻
++++++++++com/xmlmind/xmledit/dialog/Messages.properties❼
    .
    .
    .
```

❶ Beginning of first section: messages contained in the `com/xmlmind/xmledit/command/Messages.properties` resource file. As a translator, you can ignore these markers.
❷ The ID of the message is "`cancel`".
❸ English text of the message having "`cancel`" as its ID.
❹ Placeholder for the French text of the message having "`cancel`" as its ID.
❺ A comment added by the author of the English messages. As a translator, you can ignore these comments.
❻ End of first section.
❼ Beginning of next section: messages contained in the `com/xmlmind/xmledit/dialog/Messages.properties` resource file.

3.2. Second step: translate English messages to French

```
++++++++++com/xmlmind/xmledit/command/Messages.properties

cancel=Cancel
cancel=Annuler

### SelectById ---
```

```
SBI.selectId=Select ID
SBI.selectId=Sélectionner un ID

SBI.referenceToUndefinedId=Reference to non-existent ID "{0}".
SBI.referenceToUndefinedId=Référence à un ID "{0}" qui n'existe pas.
    .
    .
    .
EM.repeatAction=Repeat {0}
EM.repeatAction=Répéter {0}

----------com/xmlmind/xmledit/command/Messages.properties
++++++++++com/xmlmind/xmledit/dialog/Messages.properties
    .
    .
    .
```

Open "fr.txt" in a text editor and start filling in the blanks. Be patient, because the initial translation will take you a few hours of a pretty tedious work.

- You don't need to translate all the sections. It is sufficient to translate:

  - app_default_gui.properties

  - com/xmlmind/xmledit/command/Messages.properties

  - com/xmlmind/xmledit/dialog/Messages.properties

  - com/xmlmind/xmledit/edit/Messages.properties

  - com/xmlmind/xmledit/form/Messages.properties

  - com/xmlmind/xmledit/guiutil/Messages.properties

  - com/xmlmind/xmledit/search/Messages.properties

  - com/xmlmind/xmledit/spell/Messages.properties

  - com/xmlmind/xmledit/styledgadget/Messages.properties

  - com/xmlmind/xmledit/view/Messages.properties

  - com/xmlmind/xmleditapp/app/Messages.properties

  - com/xmlmind/xmleditapp/app/part/Messages.properties

  - com/xmlmind/xmleditapp/app/prefsheet/Messages.properties

  - com/xmlmind/xmleditapp/applet/Messages.properties

  - com/xmlmind/xmleditapp/command/Messages.properties

  - com/xmlmind/xmleditapp/dialog/Messages.properties

  - com/xmlmind/xmleditapp/docbook/Messages.properties

  - com/xmlmind/xmleditapp/kit/Messages.properties

  - com/xmlmind/xmleditapp/kit/gui/Messages.properties

  - com/xmlmind/xmleditapp/kit/part/Messages.properties

  - com/xmlmind/xmleditapp/kit/prefsheet/Messages.properties

- com/xmlmind/xmleditapp/process/Messages.properties

- com/xmlmind/xmleditapp/spreadsheet/Messages.properties

- com/xmlmind/xmleditapp/start/Messages.properties

- com/xmlmind/xmleditapp/upload/Messages.properties

- com/xmlmind/xmleditapp/urlchooser/Messages.properties

- com/xmlmind/xmleditapp/xhtml/Messages.properties

- docbook.properties

- xhtml.properties

to localize the whole GUI of XXE.

In fact, certain sections (for example, `com/xmlmind/xmledit/xsd/load` which deals with validation of W3C XML Schemas) are *very*, *very* hard to translate and it's *recommended* not to translate them.

- The translated message must be one-line long but you can fold long lines by typing a '\' immediately followed by a new line. Example:

```
badCommentChars="{0}",\n\
invalid character data in comment:\n\
cannot contain string "--" and cannot end with "-"
badCommentChars="{0}",\n\
caractères invalides dans un commentaire:\n\
ne peut contenir la chaîne "--" et ne peut se terminer par "-"
```

- The customary Java™ escape sequences are recognized by the `translatexxe` utility: '\n' (newline), '\t' (tab), '\u*XXXX*' (where *XXXX* is the code of a Unicode character in 4-digit hexadecimal notation), etc.

- The punctuation and capitalization of the original message must be respected.

- Some messages contain one underscore character ('_') (e.g. "fileMenu.label=_File" in the figure below). Such messages are menu items and the underscore is located just before the letter which is used as the menu item mnemonic.

```
### File menu ---

fileMenu.label=_File
fileMenu.label=

newAction.label=_New...
newAction.label=

openAsTemplateAction.label=Open as _Template...
openAsTemplateAction.label=

openCopyAction.label=Open Cop_y...
openCopyAction.label=

openAction.label=_Open...
openAction.label=

saveAction.label=_Save
saveAction.label=

saveAsAction.label=Save _As...
saveAsAction.label=
```

Recommendations:

- The location of the underscore depends on the language used for the menu item.

  For example: in English, **Alt-F** opens the File menu because the underscored letter is `F` (i.e. `_File`). In German, the File menu is called `Datei` and the underscored letter should be `D` (i.e. `_Datei`).

- Do not underscore the same letter twice in the same GUI component (same menu bar, same menu, same dialog box, etc).

- Because this constraint can be really tedious to check, it is better not to use this facility at all rather than using it inconsistently. *If you find underscoring menu mnemonics too tedious, do not underscore them at all*.

- A large number of messages contain strings `{0}`, `{1}`, `{2}`, etc. These are variables which are substituted by dynamically computed values just before the message is displayed to the user. The translated message must, of course, embed the same variables as the original message but the order of occurrence of the variables may be different if the syntax of the language requires so.

3.3. Third step: create the .jar file containing French messages

```
C:\Program Files\XMLmind_XML_Editor> translatexxe -j fr.jar fr.txt
```

The above command creates an easy-to-deploy `.jar` file called "`fr.jar`" out of your plain text file "`fr.txt`".

Important

If your plain text file "`fr.txt`" has not been encoded using the native encoding of your platform, you must specify its encoding using the "`-e`" option. Example:

```
C:\Program Files\XMLmind_XML_Editor> translatexxe \
    -e ISO-8859-1 -j fr.jar fr.txt
```

`Translatexxe` will guess the target language of your translation if you use a two-letter ISO code as the basename of your `.jar` file. In our example, "`fr`" is the two-letter ISO code of the French language.

Note that language variants such as "`fr-FR`" are not supported.

Tip

Nothing forces you to name your `.jar` file after the two-letter ISO code of your target language. But in such case, you need to explicitly specify to `translatexxe` which is this language by using the "`-l`" option. Example:

```
C:\Program Files\XMLmind_XML_Editor> translatexxe \
    -l fr -j French_XXE_Messages.jar fr.txt
```

As usual, if you don't remember the ISO codes, specify like "`-l foo`" and `translatexxe` will report an error and will list all valid ISO codes.

3.4. Fourth step: test your translation to French

Copy `fr.jar` to one of the two `addon/` directories scanned by XXE at startup time and start XXE.

If the language of the default locale of your platform is not the target language of your translation, you'll not be able to check what you have done. In such case, use Options|Options, General section, select the target language of your translation from the Locale combobox, then restart XXE. By doing this, you'll force XXE to use the locale you have chosen and hence, you'll be able to check your work.

3.5. When a new version of XXE is released, update your translation

Each time a new version of XXE is released, hopefully, you'll want to update your translation. For this, you need to recreate a new "`fr.txt`" out of XXE `.jar` files and your own "`fr.jar`". The command below does this:

```
C:\Program Files\XMLmind_XML_Editor> translatexxe -m fr.txt C:\temp\fr.jar
```

> ### Important
>
> Never work on your original "`fr.txt`". Always recreate a new "`fr.txt`" out of XXE `.jar` files and your own "`fr.jar`". That is, the reference here is XXE code and your "`fr.jar`". "`Fr.txt`" is just a temporary file which should be deleted after it has served its purpose.

Browse the new "`fr.txt`" and find which English messages have been added since the last time you translated XXE. These new English messages are followed by empty placeholders. Then repeat second to fourth step.

4. Command line options

```
translatexxe [-v] [-l two_letter_language_code] [-e message_file_encoding]
    -m out_message_file [in_jar_file]
OR
    -j out_jar_file in_message_file
```

- Use "`-m out_message_file`" to create a text file containing XXE messages to be translated.

- If you have already translated XXE messages, say to Italian, and want to update your translation, use "`-m out_message_file my_italian_messages.jar`".

- Use "`-j out_jar_file in_message_file`" to convert your message file "`in_message_file`" to an easy-to-deploy message jar "`out_jar_file`".

- In all cases, "`-e message_file_encoding`" may be used to specify the encoding of the input or output message file.

  By default, it is the native encoding of the platform (e.g. "`windows-1252`").

- Use "`-l two_letter_language_code`" to specify the target language of your translation of XXE messages.

  By default, this language is inferred from the basename of the message jar file. Example: "`-j it.jar msg.txt`" implies that you have translated XXE messages to Italian.

- "-v" turns verbosity on. This option is useful because it prints the number of English messages, the number of translated messages and the number of resources files containing these messages.