

# Translation HowTo

Do you speak a language that muCommander is not available in? Great! This topic describes how you can help translate muCommander in your language and by doing so, earn eternal gratitude from your peers. The instructions below sound more complicated than what they really are, so don't let them scare you!

Before you embark on this exciting endeavor, please post a short message to the [Internationalization forum](#) to let fellow users know that you're working on it, to avoid any duplicate work.

## Dictionary format

muCommander uses a file called [dictionary.txt](#) (the link points to the latest version) to localize the user interface into multiple languages. Translating muCommander into a new language consists in adding an entry for every existing key.

The format of the dictionary file is relatively straightforward but requires nonetheless some explanations. It contains a list of localized entries, expressed in the following format:

```
<key>:<lang>:<value>
```

where:

- <key> is the key used by the application to refer to the entry
- <lang> is an ISO 639-1 language code (EN, FR, zh\_CN...), full list available on [Wikipedia](#)

<value> is the text localized in the <lang> language

Here's an example:

```
warning:EN:Warning  
warning:FR:Avertissement
```

those two entries define respectively English and French translations for the `warning` key.

In addition to plain text, entry values can contain variables. There are 2 types of variables, both can be mixed with plain text:

- parameters: `%1, %2,...` Parameters are replaced at runtime by values specified by the application. For example, the following value: `Invalid path: %1` may be displayed in the application as `Invalid path: /r00t/`, should the invalid path be `/r00t`.
- references to other entries: `$(<key>)` refers to the entry named <key> and will be replaced at runtime by the value of <key> in the current language. For example, `run_dialog.stop:EN:$(stop)` will be replaced by the value of the 'stop' key, which is 'Stop' in English.

Lines that start with a '#' character are comments and are simply ignored when the dictionary file is loaded.

# Translation process

Translating muCommander into a new language consists in adding a new entry for each key defined in the dictionary file. Here are a few things you should be aware of:

- the dictionary file of the latest nightly build should be used to ensure that the list of entries is up-to-date.
- the entry keys should never be modified, only their localized values. Similarly, variable names should not be changed.
- when an entry is requested by the application in a language for which the dictionary has no defined value, the EN (English) value is used. In other words, English is the default language for untranslated entries.
- when an entry requested by the application doesn't exist in the dictionary (has no value in the requested language and no EN value), the key is used instead, allowing to spot missing values in the user interface.
- for entries containing parameters (`%1`, `%2...`): if the parameters are located at the end of the entry, you should as much as possible try and keep it that way, as the user interface sometimes relies on that. Parameters values can be of variable length and if the parameter is in the middle of the entry, the text after might get truncated if one of the parameter's values is too long.

Important: The dictionary file is encoded in the UTF-8 character encoding. Please ensure that you use UTF-8 when saving it, otherwise muCommander will not be able to load it properly.

For new languages, you'll need to add the language code to the `available_language` entry and create a `language.<lang>` entry containing the new language's name expressed in the language itself, for instance 'Français' for French. The new language should then appear in the preferences, allowing you to select it and then restart the application to see your translation.

## Testing your translation

When you're done modifying the dictionary file, you'll probably want to test it to ensure that it is working properly. Here are the steps to follow:

1. Download the latest [mucommander.jar](#) file (the link points to the latest nightly build).
2. Enter `mucommander.jar` using muCommander. You'll find the original `dictionary.txt` at the root of the JAR file.
3. Copy your `dictionary.txt` file into the JAR file (again using muCommander), overwriting the existing dictionary.
4. Start the modified `mucommander.jar` file by pressing Shift+Enter. You should see the modifications you made reflected in the user interface. Repeat this process whenever you want to test a new version of your dictionary.

Alternatively, you can edit `dictionary.txt` file directly inside the JAR file using muCommander's internal text editor. Note however that muCommander needs to be restarted each time an entry is modified for the change to become visible.

## What next?

You can post your translation to the [Internationalization forum](#) and/or send it by email at: `contributions` <at> `mucommander` <dot> `com`. We'll integrate it to the nightly build so that people can start testing it. Your translation will then be integrated to the next stable release.

Unless you specify otherwise, you'll automatically be subscribed to a (low-traffic) mailing list where we periodically ask for translation updates, usually a couple weeks before a new version of muCommander gets released. Finally, you can create an account on the bug tracker. This will allow you to be notified whenever a translation issue has been reported and resolve the issue.