Help with Translation

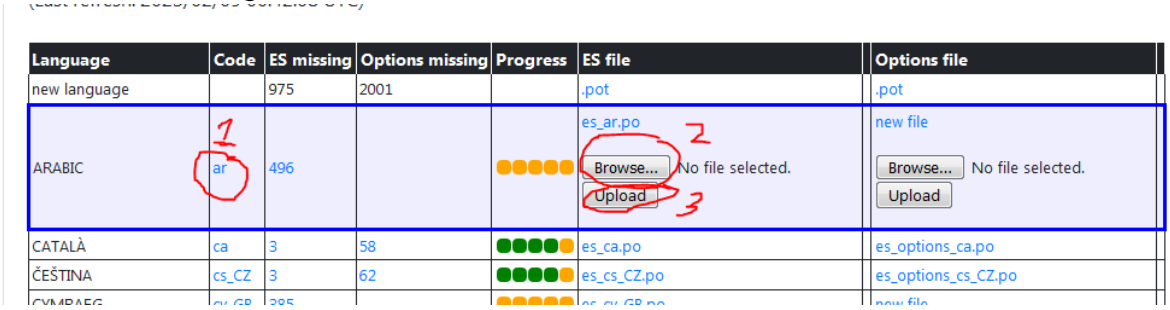
Batocera comes in a variety of local languages, and relies on the community to help with the translations. We need you for this!

For translators

1. First check if someone has already started a translation on the translations page. Download the two `*.pot` files for your language you are going to add translations for. The two files are for EmulationStation-specific strings (ES file) and system-specific options (Options file). If starting a new language, download the `*.pot` files from the new `language` row in the table.
2. Edit and then compile your translation files. You can use the Poedit program on your desktop, or online with the Poeditor tool to create the `*.po` and `emulationstation2.mo` files specific to your language.

   If using a plain text editor to make your modifications instead, make sure it keeps the Unix end-of-line terminators.

3. Test your translation to ensure no lines are cut-off in the UI, special characters appear as intended and most importantly that Batocera does not crash.
4. Once you're happy with your translation, on the translation page click the region code of your language in the “Code” column and then upload the file to the appropriate “ES file” or “Options file” column on the right side.



   If it's a brand new language, please contact the dev team on the #translations-files channel of our Discord server to get your language added to the table.

Thank you so much for your help!

- To see a list of the list of missing strings to be translated with their respective line numbers, click on the number in the **ES missing** and **Options missing** columns respectively.
- If you see a “timer” icon after the file, it means that someone just uploaded a new translation file that hasn't been merged into GitHub yet.

Test your translation

1. Merge the `emulationstation2.po` and `batocera-es-system.po` files in your preferred editor and export the `emulationstation2.mo` file from it.

2. Connect to Batocera via WinSCP (or via SSH).
3. Copy your `emulationstation2.mo` file to `/usr/share/locale/<your language code>/LC_MESSAGES/` – for instance `/usr/share/locale/pt_BR/LC_MESSAGES/emulationstation2.mo` for Brazilian Portuguese.
4. On a keyboard connected to the Batocera machine, press `[Alt] + [F4]` (or run `batocera-es-swissknife --restart` if in SSH) to refresh the language strings in ES.
5. Now explore ES to view all the strings you've translated!

To keep your changes persistent between reboots, connect via SSH if you haven't already and run `batocera-save-overlay` to save the overlay.

For developers

Manually add in a line to be translated

If there is a line in EmulationStation that does not have a string entry in the PO file, then it can be manually added in with the fake_gettext library:

```
#define fake_gettext_stringname     pgettext("category", "string")
```

For example:

```
#define fake_gettext_flatten_glow           pgettext("game_options",
"FLATTEN-GLOW")
```

Place this into the top of the function which contains the untranslated string and upon next compilation it will be added to all the relevant PO files.

Add a comment in the code for translators

If you are coding something new in Batocera and think that the string to be translated might be a bit ambiguous for translators (remember, they will only see the string in isolation, which might not make sense outside of its context), you can add a comment just for translators to see when they get to that string. In EmulationStation or Batocera's es_features, add the following comment right before the code in which the string appears:

```
/* TRANSLATION: Short comment that explains the context of the string. */
```

For example, this is something that has been added to `batocera-emulationstation/es-app/src/guis` in the source code:

```
/* TRANSLATION: For general system information, related to the machine. */
informationsGui->addGroup(_("INFORMATION"));
```

Blacklisted words

What if you want to add in a string that's explicitly not meant to be translated, such as a proper noun, technical code string or the model name of a retro system?

Most of these kinds of strings should be automatically picked up and removed by the `createEsTranslations` function in https://github.com/batocera-linux/batocera.linux/blob/master/package/batocera/emulationstation/batocera-es-system/batocera-es-system.py, however there might come a case where a word fails to fall into one of its filters.

Such words can be manually appended to https://github.com/batocera-linux/batocera.linux/blob/master/package/batocera/emulationstation/batocera-es-system/locales/blacklisted-words.txt in the source code. Just add the word on a new line, simple as that.

What if I wanted to edit an English string?

Generally editing already-existing English strings in the CPP files is discouraged, as it will remove the translation that was made for it (if it exists). However, there may be times when the English string is misspelled or just plain incorrect and needs to be amended, despite it already being translated.

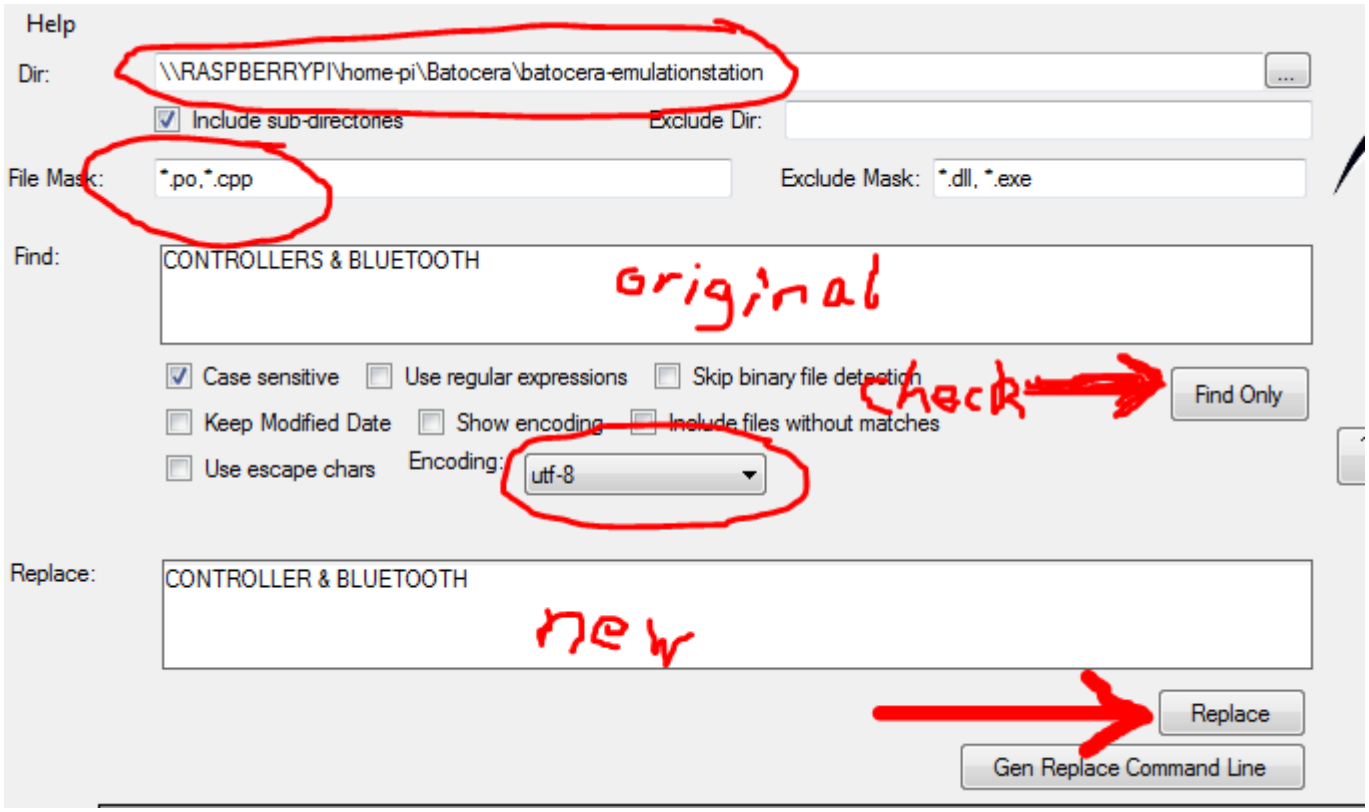
The best way to go about this is to compile batocera-emulationstation, as that will automatically run the `xgettext` and `msgmerge` submodules used to generate the `emulationstation2.pot` and subsequent PO files. You can then use a program like POEdit or just a plain text-editor to restore the translations (old translations are moved to being comments).

Understandably, this is a lot of work, especially with how many languages Batocera supports. So here's a mini-tutorial on how to automate the process (on Windows):

This method does cause corruption when re-imported back into POEdit, so double check that first before re-merging.

1. Download and install the Find and Replace tool and Notepad++.
2. Run `fnr.exe` from the ZIP file.
3. Set the **Dir** to the root of the `batocera-emulationstation` folder.
4. Add `*.po` to your **File Mask**. If you want to also use this tool to edit the string in the CPP itself, you can instead use `*.po,*.cpp`.
5. Set **Encoding** to `utf-8`. Note that this isn't actually the correct encoding, but it's close enough.
6. Put in the original string you want to replace into the **Find** textbox.
7. Click **Find Only** to test to see which strings it finds. This is important, especially if also editing CPP files, as it ensures you aren't accidentally overwriting code. If you want to be safe, enclose your find text in double-quotes (“”).
8. Put in the new string you want to replace it with into the **Replace** textbox.
9. Click **Replace** (the button… not the textbox label).

If you aren't sure of the setup, here's a professional grade pictograph detailing the settings that have been changed from default:



Since you've already cloned batocera-emulationstation, you can just push the PR to merge it. But what's this? You've noticed that after running `git diff` (as every good developer does before merging a commit I'm sure), every file you've edited now has an additional, invisible character at the beginning that you're sure you didn't insert yourself? That's because Find and Replace is actually not so good with encoding text files so it inserts the Windows 16-bit signature into all the UTF-8 text files at the beginning. This is easy enough to fix.

1. Open the afflicted text file with Notepad++.
2. Click **Encoding** → **UTF-8**. You'll notice that it was originally on “UTF-8-BOM”.
3. Save the file.
4. Rinse and repeat for all afflicted text files.

This should get rid of the extra character at the beginning of the text files.

From:
https://wiki.batocera.org/ - **Batocera.linux - Wiki**

Permanent link:
https://wiki.batocera.org/help_with_translation

Last update: **2024/08/16 01:03**