AMuseWiki

Localization

Localization

amusewiki.org

Contents

If you need to update the translations in the UI, you have two ways to do that, globally or locally for a single site.

Global

GNU gettext message catalogues (.po files) are located at `lib/AmuseWikiFarm/I18N/xx.po`, where xx is the language code. Changes done here shouldn't be site specific and if you improve the translations, please send the updated .po file to me (melmothx@gmail.com) or fork the amusewiki repo and do a pull request, making sure you're aligned to the latest master branch.

Local `PO` files

You can override the UI translations and provide them for the categories as well (if your site is multilingual or if you use codes instead of strings) storing the relevant po files into the `site_files/locales` directory in the site repository (e.g. `repo/amw/site_files/locales/hr.po`).

`lexicon.json`

Before version 1.8, the only way to override the translation was using a JSON file in the `site_files/lexicon.json` file.

Internally, the application will see only the PO files. However, depending on your needs and preferences, you may find less complicated to continue to use this file instead of the po files.

On application restart, these files are read and PO file produced (or when you call amusewiki-upgrade-lexicon against the repo directory.

So, you have two alternative ways:

- leave the po files uncommitted in the git repo, and continue as before (keeping in mind that you need to manually restart the app or call `amusewiki-upgrade-lexicon` to produce the po files when you update it).
- commit the po files in the git and remove `lexicon.json`, banishing it.

Mixing the two approaches will just lead to confusion, git conflicts, and so on.

The format is as follows (plain JSON):

```
{
   "term" : {
      "hr" : "Term in croatian",
      "en" : "Term in english",
      "it" : "Term in italian"
   },
   "another term [_1]" : {
      "hr" : "Another term in croatian with an argument [_1]",
      "en" : "Another term in english with an argument [_1]",
      "it" : "Another term in italian with an argumetn [_1]"
   }
}
```

The JSON must be correct, otherwise you will not get any translation out of that. You can use arguments for translations with the `[_1]`, `[_2]` syntax or the gettext syntax (%1, %2, etc.).

Strings are expected to be unescaped (HTML-wise).

To validate the lexicon file prior to upload, use this (somehow longish) one-liner:

```
perl -MJSON -MData::Dumper \
     -e 'open (my $fh, "<:encoding(UTF-8)", "site_files/lexicon.json
         $/ = undef; my $json = <$fh>;
            print Data::Dumper::Dumper(JSON::from_json($json));'
```

You can create the po files out of those files with the script `amusewiki-upgrade-lexicon`. E.g.

```
script/amusewiki-upgrade-lexicon repo/*
```

Which will parse the lexicon.json files and dump the po files, merging them if already existing. No git operation is performed.

Adding a new language (developer-only)

Go to lib/AmuseWikiFarm/I18N and run (for example to add swedish)

```
msginit --input=messages.pot --output=sv.po --locale=sv
```

Add the language code and the language string to the `AmuseWikiFarm::Utils::Amuse::known_langs` method.

Add the language code to the `script/upgrade_i18n` script.

Add the babel name to `Text::Amuse::_language_mapping` if missing.

Tests which have a plan depending on the number of supported languages:

- `t/controller_Latest.t`
- `t/multilang.t`