ViMP 3.5

Adding a new language

Author:
ViMP GmbH

Table of Contents

About this document

This document describes how to add additional languages to a ViMP installation. As an example we add a Spanish translation.

At http://www.vimp.com/en/free-downloads.html we offer several language packages. If your desired language is not available, please consider that you have to do the translation by yourself. This document only describes how to add a language to the system.

Adding language settings

With ViMP 3.0 adding a language became significantly easier than before.

All you have to do is to run the following command in the base directory of your ViMP installation:

```
./symfony i18n:create
```

Next, follow the instructions on the screen. You will be prompted to enter the following parameters:

- English name of the language
- native name of the language
- ISO 639-3 language code (http://en.wikipedia.org/wiki/List_of_ISO_639-1_codes)
- write direction (left-to-right or right-to-left)

See an example output in the following:

```
>> i18n      culture data
English name: Spanish
Native name: Espanol
ISO 639-3 code: spa
Direction [ltr]:
>> i18n      copy languages files
>> i18n      create culture catalogue
>> i18n      language: spanish
>> i18n      importing i18n strings from
“plugins/stLanguageSpanishPlugin/i18n/messages.xml"
>> i18n      importing i18n strings from
“plugins/stLanguageSpanishPlugin/apps/backend/i18n/messages.xml"
>> i18n      importing i18n strings from
“plugins/stLanguageSpanishPlugin/apps/frontend/i18n/messages.xml"
```

If you don’t want to import the contents during creation, but only want to create the structure, add the parameter --no-import:

```
./symfony i18n:create --no-import
```

For importing the XLIFF files into ViMP at a later date, execute the following command:

```
./symfony i18n:import
```

This command imports all new XLIFF file records into the database.

Optionally a language code can be added, if you want to import one language only:

```
./symfony i18n:import es
```

Please note to use the ISO 639-1 code of the language here.

For more options, please consult the integrated help.

If you visit your ViMP portal at that time, you should already see Spanish as a new option in the language select box. You can also switch the language yet. But you’ll see a lot of English words at that moment. Let’s get back to this later.

Translations in ViMP

Before adding your own translation we want to tell you a bit more about the functioning of the translation system in ViMP and where you can add your own texts.

The underlying symfony framework already comes with a lot of internationalization (i18n) and localization (l10n) functions. There is a complete set of translation classes that are used by ViMP as well. The translation always begins with the source text, which is English in ViMP. The symfony framework catches this text, checks if there is a translation in the chosen language, and outputs the translation if available. If no translation was found, the source text will be displayed.

The languages supplied with ViMP conform to the XLIFF catalogue format (see Appendix: XLIFF in a nutshell) and will be imported into the database automatically during the initialization of ViMP. With ViMP the XLIFF files only act as a simple way to manage translations and as an exchange format between ViMP and miscellaneous translation tools. Actually the translations will always be read from the database during operation.

Translations within the XLIFF files can exist at different levels: project-wide, application-wide and module-wide. The more specific one always overwrites the less specific one. ViMP makes use of project-wide and application-wide translations as well as additional translations in different plug-ins.

If you take a look at the translations, you'll find some texts containing words with a prefixed @:

```
Contact @user
```

Above, `@user` is a placeholder. If the translation is requested by the system, a value will be passed back that replaces `@user`. This enables you to translate your texts grammatical correct without worrying where the data will be inserted. Assure to keep the placeholders working to avoid fragmented and disrupted texts.

Creating translations

There are two ways to translate ViMP. You can either use the backend integrated tool to edit the translations, or you edit the XLIFF files directly with a suitable editor (see Appendix: XLIFF in a nutshell) and import them with the command line tool.

Translate each entry into the new language. Save some time for the translation, because there are about 3000 strings and fragments to translate. Don't expect the translation to be done in a couple of minutes.

If you decide to translate ViMP using the XLIFF files: never change the text within the `source` tags. The translation class searches exactly for the given wording. Even the smallest modification (e.g. the correction of a typo) results in an error, because ViMP cannot find the text any more. The list of files that have to be translated is as follows:

- `plugins/stLanguageSpanishPlugin/i18n/messages.xml`
- `plugins/stLanguageSpanishPlugin/apps/frontend/i18n/messages.xml`
- `plugins/stLanguageSpanishPlugin/apps/backend/i18n/messages.xml`
- `plugins/stLanguageSpanishPlugin/apps/webtv/i18n/messages.xml`
- `plugins/stLanguageSpanishPlugin/plugins/*/i18n/messages.xml`

In order to save the XLIFF modifications to the system, they have to be reimported into the database. Execute the following command in the base folder of your ViMP installation:

```
./symfony i18n:import --replace
```

Attention: This command overwrites all changes that have been done via the backend translation tool.

You can also add the language code here to import only one language:

```
./symfony i18n:import --replace
```

If you translated ViMP via the backend you can back up your changes into the existing XLIFF files. Execute the following command:

```
./symfony i18n:export
```

With the following command it is possible to export a single language:

```
./symfony i18n:export es
```

While testing your translations you should bear in mind that they will be cached for one day by default. In order to see the latest changes, you should empty the i18n cache:

```
./symfony cc --type=i18n
```

The i18n cache will be emptied, but the configuration and template caches will be kept.

When you‘re done

After you finished all steps the new language can be used. But consider that new versions of ViMP can contain new texts or existing texts might be modified or removed. So before updating your ViMP portal always check the translations and assure that your translations are up-to-date and complete.

If you want to share your translation, please contact us via e-mail at **info@vimp.com** to work out the details.

Appendix

XLIFF in a nutshell

As already mentioned, ViMP makes use of the XLIFF catalogue format. This chapter explains how to write XLIFF files.

What are XLIFF files?

XLIFF stands for **X**ML **L**ocalisation **I**nterchange **F**ile **F**ormat and is a multilingual file exchange standard for localization, adopt by the OASIS. This format makes it possible to translate complete applications like e.g. ViMP.

Where do I find the XLIFF files inside ViMP?

The XLIFF files follow the pattern `messages.xml` and are located within the language plugin folder (e.g. /plugins/stLanguageSpanishPlugin).

The main XLIFF files are located in the `i18n` folders within the language plugin folder and the application folders, e.g. `i18n/messages.xml` or `apps/frontend/i18n/messages.xml`.

Additionally, every plugin can contain its own set of XLIFF files. They are located in the `i18n` folders within the plugin folder, e.g. `plugins/stStatisticsPlugin/i18n/messages.xml`.

What does a XLIFF file look like?

Let's start with an empty XLIFF file:

```
<?xml version="1.0" ?>
<!DOCTYPE xliff PUBLIC "-//XLIFF//DTD XLIFF//EN" "http://www.oasis-
open.org/committees/xliff/documents/xliff.dtd">
<xliff version="1.0">
  <file source-language="en" target-language="es" datatype="plaintext"
original="messages" date="2009-03-02T12:48:00Z">
    <header />
    <body>
    </body>
  </file>
</xliff>
```

The highlighted part is where the translation is defined. The `source-language` and `target-language` settings contain the ISO 369-1 codes for the translation languages. ViMP always uses the source language `en`. As we want to add a Spanish translation, the target language is `es`.

Each translation string is defined as a `trans-unit` block inside the `body` tag:

```
<trans-unit id="1">
  <source>Sprache</source>
  <target>Idioma</target>
</trans-unit>
```

The ID of each block has to be unique within the XLIFF file. The `source` tag contains the original text, the `target` tag contains the translation.

This is how the complete file could look like:

```
<?xml version="1.0" ?>
<!DOCTYPE xliff PUBLIC "-//XLIFF//DTD XLIFF//EN" "http://www.oasis-
open.org/committees/xliff/documents/xliff.dtd">
<xliff version="1.0">
  <file source-language="en" target-language="es" datatype="plaintext"
original="messages" date="2009-03-02T12:48:00Z">
    <header />
    <body>
      <trans-unit id="1">
        <source>Sprache</source>
        <target>Idioma</target>
      </trans-unit>
    </body>
  </file>
</xliff>
```

How to modify a XLIFF file

1. With a XLIFF editor

There are various commercial and non-commercial editors and translation tools on the market that can handle XLIFF files. You should take account of the fact that the editor must be able to save the XLIFF files in the UTF-8 format with UNIX end-of-lines.

An example for a platform independent translation tool, written in JAVA, are the **Open Language Tools**.

2. Manually

When you modify XLIFF files with a normal text editor, again, you have to consider that the editor must be capable to save the XLIFF files in UTF-8 format with UNIX end-of-lines.

Additionally the `source` tags must not be modified in any way, because ViMP won't be able to relate the translations inside the `target` tags otherwise.