Table of contents:

1 - Needed files and tools:

Creating and/or maintaining translations do not need any skill in C++ programming: **there is no change to do in Kicad files**.
Translations are easy to do with a tool **PoEdit** that locate (in Kicad sources) sentences to translate and is able to create a dictionary for Kicad from translations created with this tool.
So you need to install PoEdit, and get latest Kicad sources, and, for existing translations, get latest translations.
Translations can be made under Linux, Window or MacOSX

1.1 - Download PoEdit

See http://www.poedit.net/

1.2 - Downloading Kicad sources:

Kicad sources are currently (july 2011) hosted by Launchpad.
(See https://launchpad.net/kicad)
Files can be downloaded from Launcpad by using a tool named "**bazaar**" (**bzr** in commands).
So:

- Install (if not already done) the tool named bazaar (easy to install under all platforms) : see http://bazaar.canonical.com/
- Download Kicad sources using the command
  bzr co lp:kicad <directory where sources files are copied>
- You'll find this doc about translation and poedit configuration in Documentation: see Documentation/GUI_Translation_HOWTO.pdf

1.3 - download existing translations and documentations

Kicad translations and documentations are currently (july 2011) also hosted by Launchpad.https://code.launchpad.net/~kicad-developers/kicad/doc
Download translations using command:
bzr co lp:~kicad-developers/kicad/doc <directory where doc files are copied>

2 - Find sentences to translate:

The different menus and tool tips in KiCad are internationalized, and can be easily translated into a local language ***without source code modifications***.
The rules are:

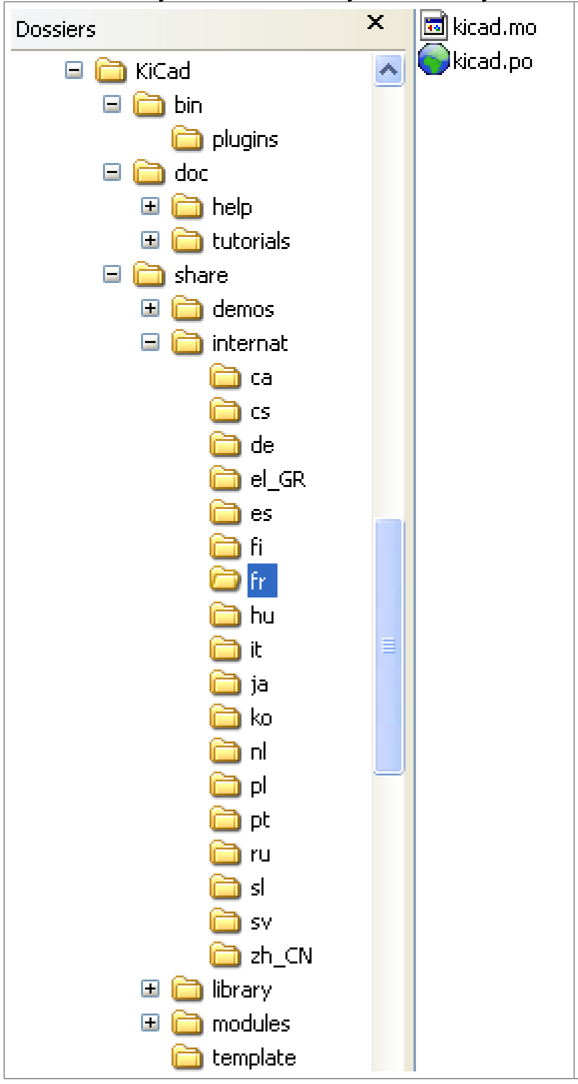
- They are written in english.
- All strings which must be translated are written like: **_("hello world")**, and displayed "hello world" but if a dictionary is found translated into the locale language before displaying.
- A dictionary English***->locale*** handle translation (one dictionary by language).

The easier way to create and maintain the dictionary English***->locale*** is to use, ***poedit***. (www.poedit.net).
Poedit scans KiCad sources and allows you to enter translations.
You must download KiCad sources and set poedit in order to create translations.

3 - KiCad tree for translations:

3.1 - Dictionary tree:

The dictionary will be found by KiCad only if it is in a suitable path:



The suitable path is ***kicad/internat/xx***, or ***kicad/internat/xx_yy***
with: ***xx*** = normalised locale indicator (short form) like:

- fr = france
- en = english
- es = spanish
- pt = portuguese

or: ***xx_yy*** = normalized locale indicator (long form) like:

- fr_FR
- en_GB
- en_US

3.2 - Search path:

Dictionaries and on-line help files are searched in this order:

- In the path in normalized locale indicator (long form) (kicad/internat/xx_yy)
- In the path in normalized locale indicator (short form) (kicad/internat/xx)

And for on-line help files search is made in

- In the path in normalized locale indicator (long form) (kicad/help/xx_yy)
- In the path in normalized locale indicator (short form) (kicad/help/xx)
- kicad/help/en
- kicad/help/fr

Note:
The main KiCad path in retrieved from the binary path, or (if not found):
under windows:

- c:\kicad
- d:\kicad
- c:\Program Files\kicad

or under linux:

- /usr/share/kicad
- /usr/local/share/kicad
- /usr/local/kicad/share/kicad
- /usr/local/kicad

3.3 - Files:

In each directory there are 2 files ***kicad/internat/xx***:

- internat.po (the dictionary file
- internat.mo (the poedit work file)

4 - Using poedit

4.1 - Installation

Download and install poedit (www.poedit.net).
Poedit exists for Windows, Linux and Mac OS X.

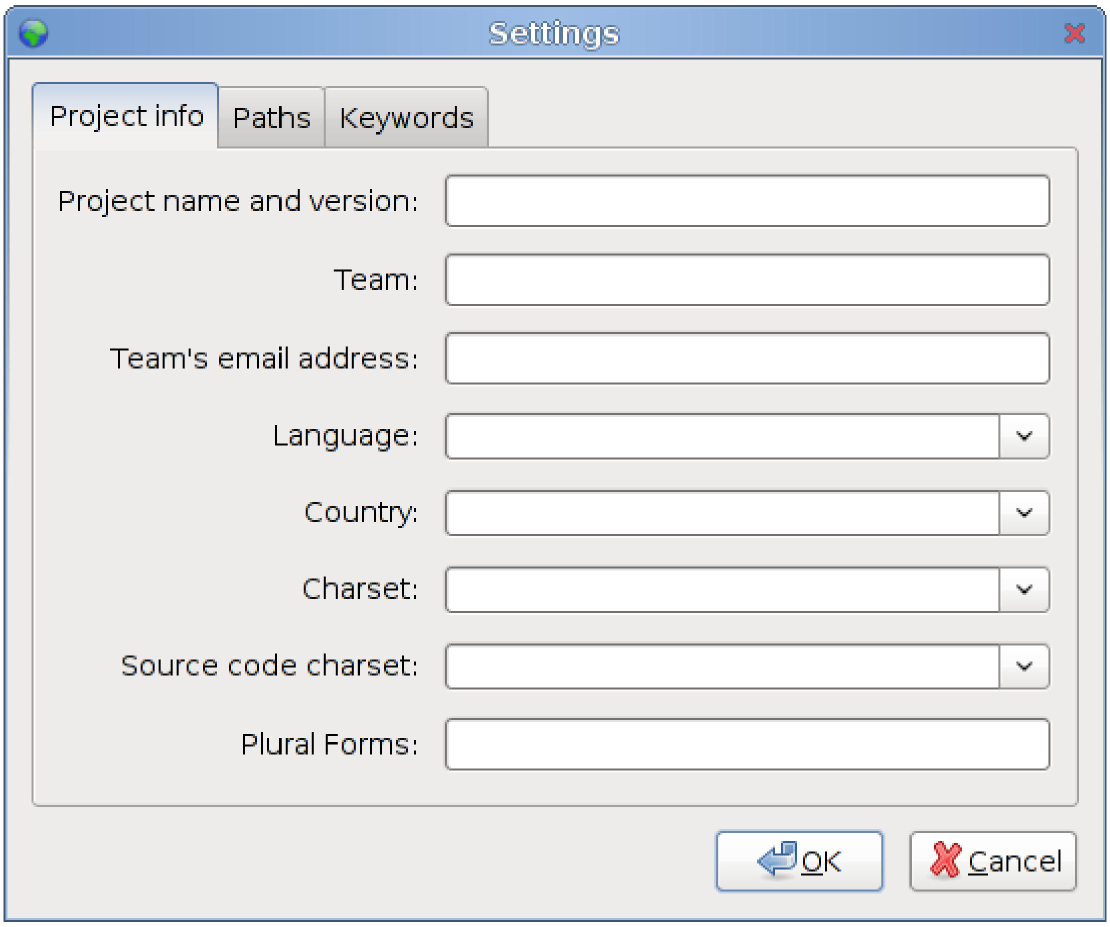
Download and unzip KiCad sources.

4.2 - KiCad preparation

KiCad sources: in this example files are in f:/kicad/.
All the strings to translate are tagged like ***_("string to translate")***.
poedit must search the _ (underscore) symbol to locate these strings.
One must add in KiCad the suitable directory for the dictionary (***kicad/share/internat/xx)***.
In this example, the directory is ***kicad/share/internat/fr.***

4.3 - Poedit Configuration

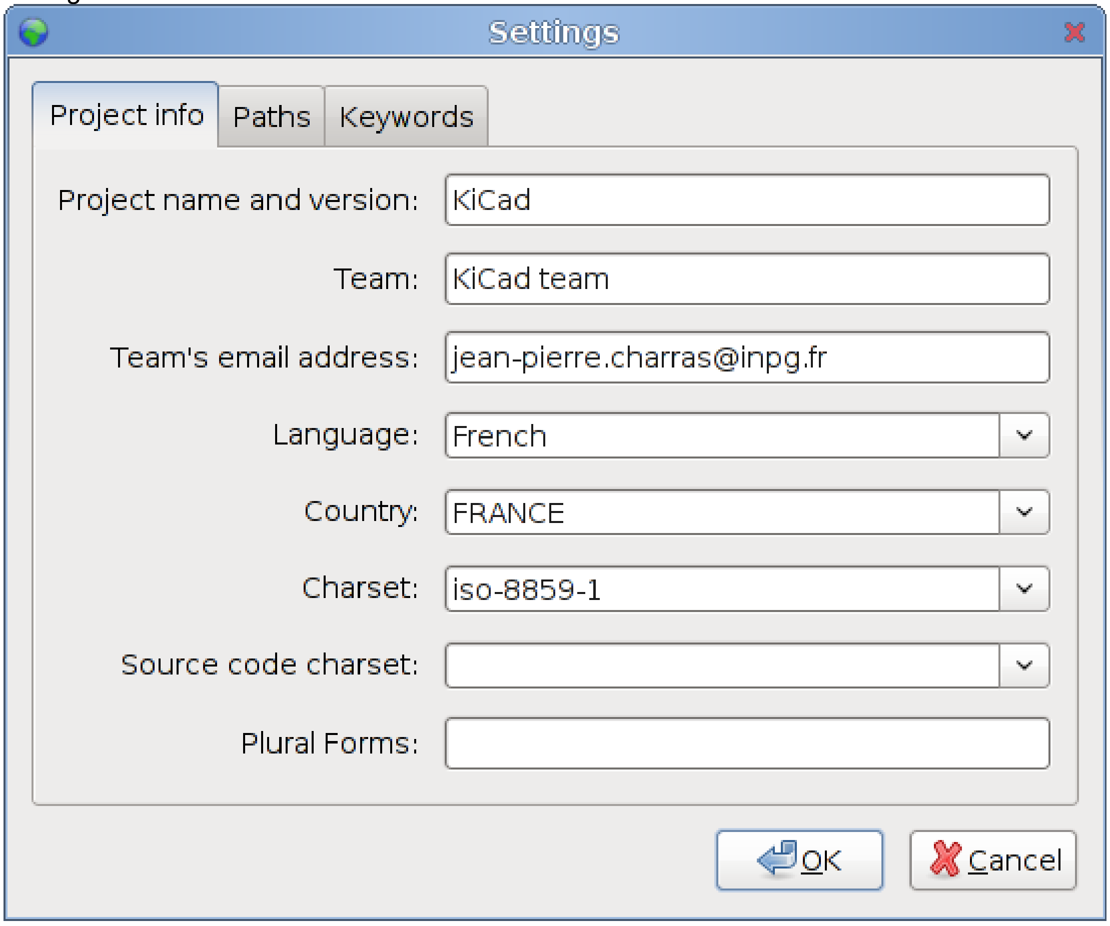
Run poedit.
Run ***File/New catalog...***
You should see something like:



Project name and version:	
Team:	
Team's email address:	
Language:	
Country:	
Charset:	
Source code charset:	
Plural Forms:	

4.4 - Project Configuration:

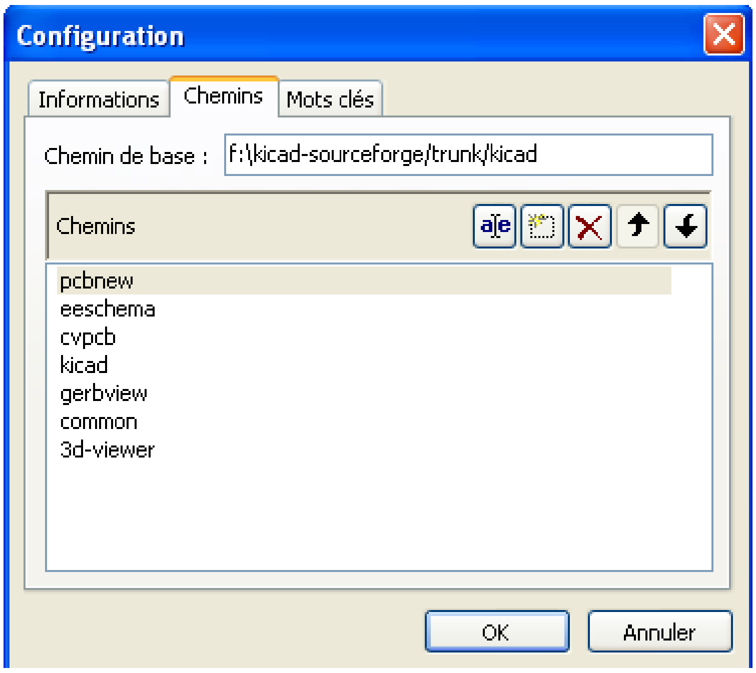
Here is the configuration for the French translation:



Project name and version:	KiCad
Team:	KiCad team
Team's email address:	jean-pierre.charras@inpg.fr
Language:	French
Country:	FRANCE
Charset:	iso-8859-1
Source code charset:	
Plural Forms:	

The source files are in English, so no need to choose something for source code.

4.5 - Path and files Configuration:

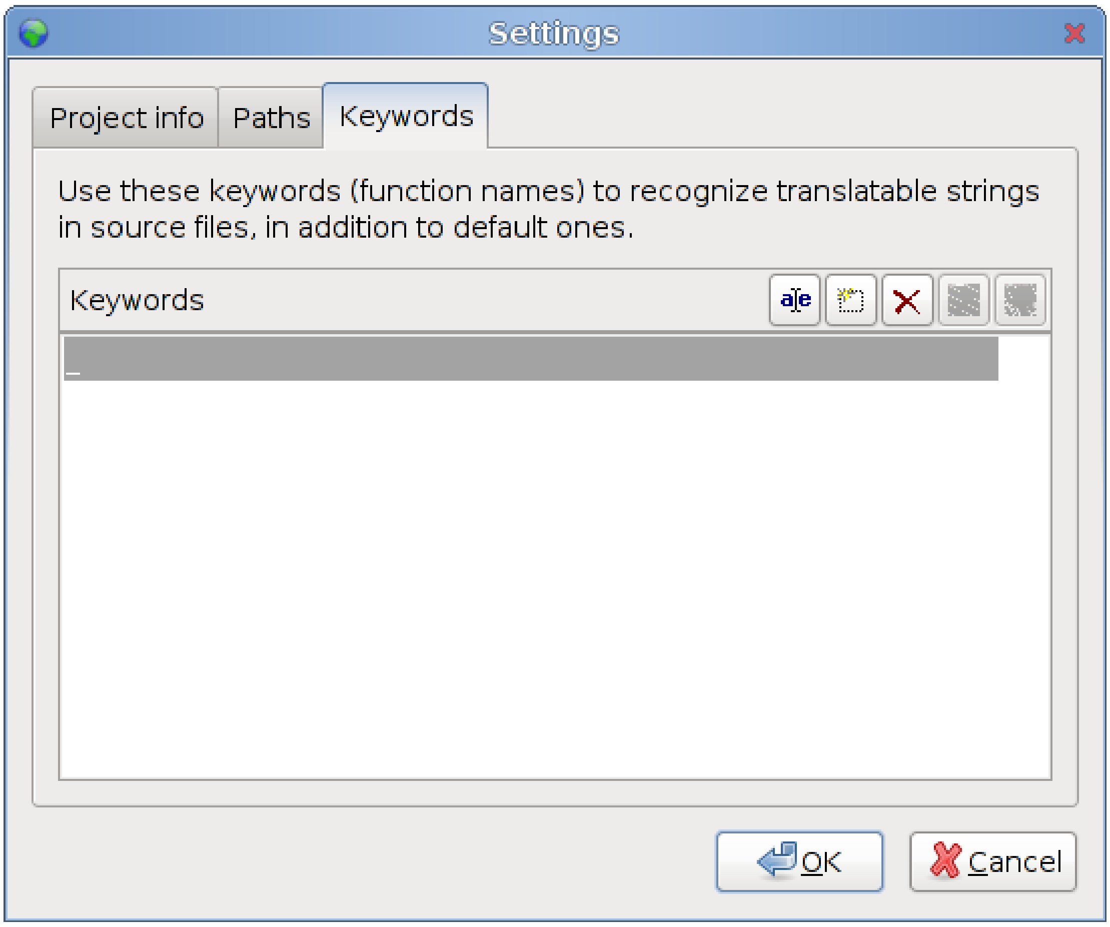

Configuration

Chemin de base : f:\kicad-sourceforge/trunk/kicad

Chemins

pcbnew
eeschema
cvpcb
kicad
gerbview
common
3d-viewer

4.6 - Keyword Configuration:


Settings

Use these keywords (function names) to recognize translatable strings in source files, in addition to default ones.

Keywords

_

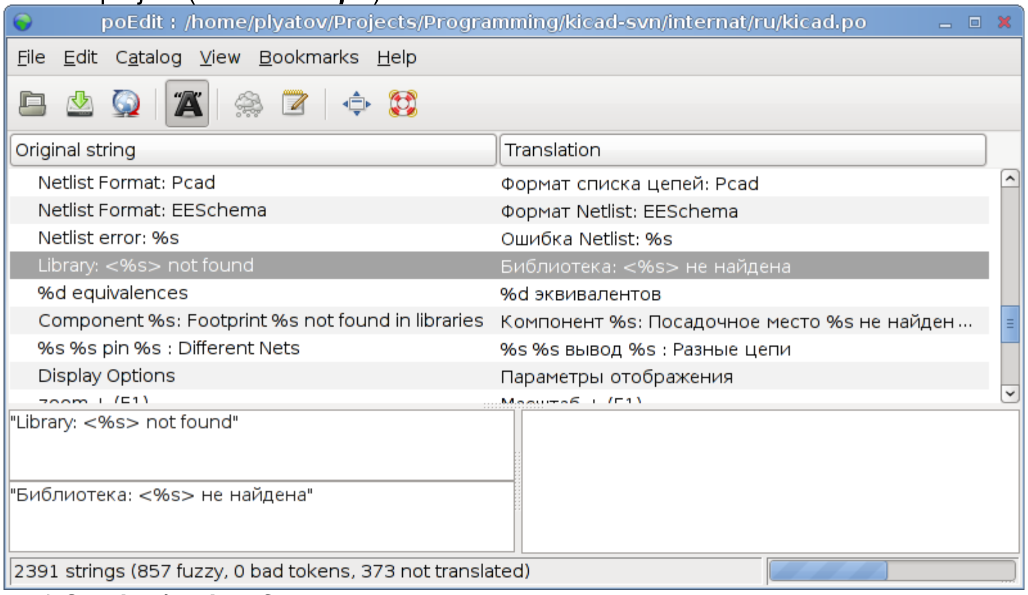
Only one keyword to enter: _ (underscore) used as tag in source files

4.7 - Save the project:

Save the new projet in ***kicad/share/internat/xx*** with the name ***kicad.po***

5 - Create or edit a dictionary:

Run poedit and load a project (here: ***kicad.po***).



poEdit : /home/plyatov/Projects/Programming/kicad-svn/internat/ru/kicad.po

Original string	Translation
Netlist Format: Pcad	Формат списка цепей: Pcad
Netlist Format: EESchema	Формат Netlist: EESchema
Netlist error: %s	Ошибка Netlist: %s
Library: <%s> not found	Библиотека: <%s> не найдена
%d equivalences	%d эквивалентов
Component %s: Footprint %s not found in libraries	Компонент %s: Посадочное место %s не найден ...
%s %s pin %s : Different Nets	%s %s вывод %s : Разные цепи
Display Options	Параметры отображения

"Library: <%s> not found"

"Библиотека: <%s> не найдена"

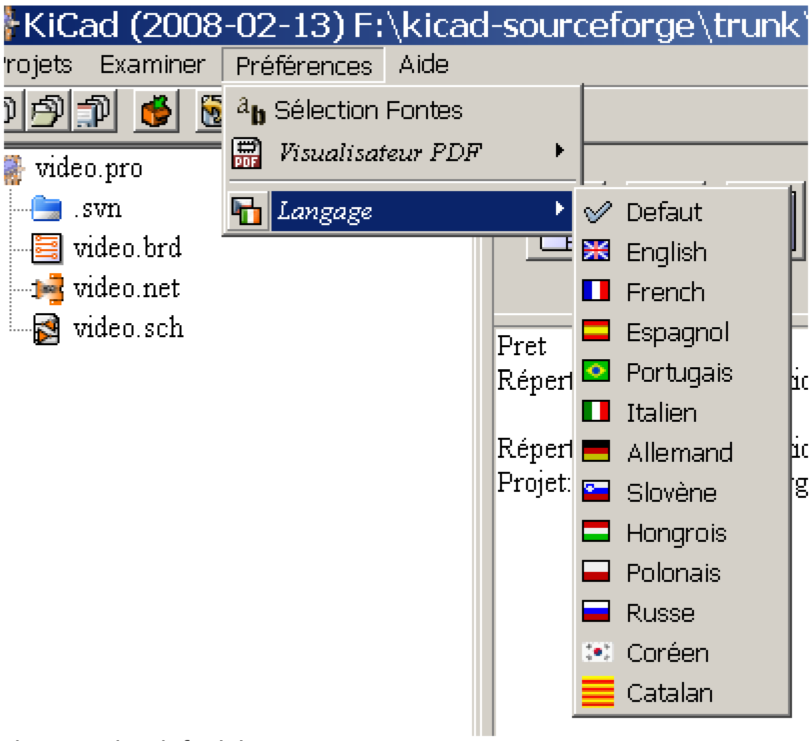
2391 strings (857 fuzzy, 0 bad tokens, 373 not translated)

Run the command ***Catalog/update from sources***.
New strings (not yet translated) will be displayed on the top of the window.

6 - Adding a new language entry in KiCad source code

This step in NOT required.
It is usefull only for developers, and for testing purpose only

In KiCad we can force the used language.



It is highly recommended to use the default language.
But because developers must test translations, a new entry in the language list can be useful for testing purposes.

6.1 - Steps:

6.1.1 - Adding a new id in include/id.h.

- In *include/id.h*, locate the sequence like:

```
ID_LANGUAGE_CHOICE,
ID_LANGUAGE_DEFAULT,
ID_LANGUAGE_ENGLISH,
ID_LANGUAGE_FRENCH,
ID_LANGUAGE_SPANISH,
ID_LANGUAGE_GERMAN,
ID_LANGUAGE_RUSSIAN,
ID_LANGUAGE_PORTUGUESE,
ID_LANGUAGE_ITALIAN,
ID_LANGUAGE_SLOVENIAN,
ID_LANGUAGE_HUNGARIAN,
ID_LANGUAGE_POLISH,
ID_LANGUAGE_KOREAN,
ID_LANGUAGE_CATALAN,
ID_LANGUAGE_UNUSED3,
ID_LANGUAGE_UNUSED4,
ID_LANGUAGE_CHOICE_END,
```

and add a new entry in list (which will be used later in menus) like:
ID_LANGUAGE_MY_LANGUAGE (one can replace a line like ID_LANGUAGE_UNUSED3 if exists to do that).

6.1.2 - Adding a new icon (aesthetic purpose only)

- Create a new icon in xpm format: usually the country flag.

Others language icons are in *common/bitmaps*
This is text like:

```
/* XPM */
static const char * lang_fr_xpm[] = {
"16 16 5 1",
"     c None",
".    c #000000",
"+    c #0000D2",
"@    c #FFFFFF",
"#    c #F00000",
"                ",
"                ",
"..............  ",
".++++@@@@####.  ",
".++++@@@@####.  ",
".++++@@@@####.  ",
".++++@@@@####.  ",
".++++@@@@####.  ",
".++++@@@@####.  ",
".++++@@@@####.  ",
".++++@@@@####.  ",
".++++@@@@####.  ",
"..............  ",
"                ",
"                ",
"                "};
```

This is a new file like *common/bitmaps/Lang_My_language.xpm*, starting by something like:

```
/* XPM */
static const char * lang_my_language_xpm[] = {
"16 16 5 1",
"     c None",
```

6.1.3 - Editing common/edaappl.cpp

- locate the text:

```
#ifdef __WINDOWS__
/* Icons for language choice (only for Windows)*/
#include "Lang_Default.xpm"
#include "Lang_En.xpm"
#include "Lang_Es.xpm"
#include "Lang_Fr.xpm"
#include "Lang_Pt.xpm"
#include "Lang_It.xpm"
#include "Lang_De.xpm"
#include "Lang_Sl.xpm"
#include "Lang_Hu.xpm"
#include "Lang_Po.xpm"
#include "Lang_Ko.xpm"
#include "Lang_Ru.xpm"
#include "Lang_Catalan.xpm"
#endif
```

and add a line to include the new icon file.

- Locate:

```
struct LANGUAGE_DESCR
{
    int           m_WX_Lang_Identifier;                  // wxWidget locale
identifier (see wxWidget doc)
    int           m_KI_Lang_Identifier;                  // kicad identifier used in
menu selection (see id.h)
    const char**  m_Lang_Icon;                           // the icon used in menus
    const wxChar* m_Lang_Label;                          // Label used in menus
    bool          m_DoNotTranslate;                      // set to true if the
m_Lang_Label must not be translated
};

#define LANGUAGE_DESCR_COUNT 14
static struct LANGUAGE_DESCR s_Language_List[LANGUAGE_DESCR_COUNT] =
{
    {
        wxLANGUAGE_DEFAULT,
        ID_LANGUAGE_DEFAULT,
        lang_def_xpm,
        _( "Default" )
    },
    {
        wxLANGUAGE_ENGLISH,
        ID_LANGUAGE_ENGLISH,
        lang_en_xpm,
        wxT( "English" ),
        true;
    },
    {
        wxLANGUAGE_FRENCH,
        ID_LANGUAGE_FRENCH,
        lang_fr_xpm,
        _( "French" )
    },
```

and add a new entry like:

```
    {
        wxLANGUAGE_MY_LANGUAGE,
        ID_LANGUAGE_MY_LANGUAGE,
        lang_my_language_xpm,
        _( "My_language" )
    },
```

wxLANGUAGE_MY_LANGUAGE is the wxWidgets language identifier for the country (see wxWidget doc).

6.1.4 - Recompiling

Obviously, this is the next and last step.