Information for developers of *easycam*

© Michael Strecke, mstrecke@users.berlios.de

Table of Contents

Revisions:

2005-12-28:

- glade, gtranslator

2005-12-31

- added dependency list
- gettext, intltool-update -d

Packet dependencies

- python-gnome2
- python-glade2
- gcc-3.4
  Most kernels are still compiled with this version of gcc, therefore the kernel modules must use it, too
- make
- ??? binutils

Easycam configuration file

The configuration file is an XML file with the following structure

Element	***Description***
easycam	root element
driver	e.g. spca, pwl
camera	for each camera that the driver supports
cameraname	name of that camera, e.g. Logitech, Inc. QuickCam Express
usbid	USB ID of that camera, 046d:0920
camera	next camera
...	...
driver	next source code, e.g. ov511
...	...

Element description

easycam

root element

attributes:

easycam.date	creation date of this config file
easycam.url	url for newer config file

driver

for each source code driver (e.g. spca)

attributes:

driver.name	name of the driver (e.g. spca), derived from file name
driver:needs	minimal version of easycam needed to install the source of this driver
driver:homepage	URL of project homepage for this source code
driver.source	URL for the current source code

camera

camera that the source code of the parent can decode

cameraname

camera name of the parent camera

usbid

usbid of the parent camera

id2xml.py

This utility compiles various text files into the easycam config file (easycam.xml).

URL of config file

This URL can be used to download newer version of the easycam config file. This information is hard coded in **id2config.py**:

```
easycamurl = "http://blognux.free.fr/easycam/latestconfig.xml"
```

All other information is taken from the files in **./cams** which have the following structure

Example: spca

```
# Comments live here
homepage http://mxhaard.free.fr
source http://mxhaard.free.fr/spca50x/Download/spca5xx-20051212.tar.gz
needs 3.0

ID 046d:0920 Logitech, Inc. QuickCam Express
ID 0ac8:301b Z-Star Microelectronics Corp. Sansun SN-510 WebCam
ID 04fc:0561 Sunplus Technology Co., Ltd
ID 041e:4028 Creative Technology, Ltd
...
```

All files in ./cams are scanned (remove backup files (.bak, .rv, ~...).

Update strategy

(for later implementation)

Update of the config file itself

The root element of the config file (easycam) has the following attributes: *easycam.date* and *easycam.url*

easycam.url is used to download a potentially newer version of the config file
easycam.date is then used to determine if the file is more recent.

Needed version of easycam

In order to install a driver easycam runs two scripts: the *easycam script* and the *driver script*.

The *easycam script* creates symlinks, apt-get-s the kernel source, etc. and runs the driver script.
The *driver script* (which is part of the driver tarball) does the actual compiling.

The attribute *driver:needs* of the element *driver* will be updated when the **easycam script** for this driver changes.

Typical situation:

- The driver xyz can be installed with easycam 10.0 :-) -> *driver.needs* = 10.0
- We find a bug in the easycam script for driver xyz.
- We correct the bug, and release easycam 10.1.
- In the config file for this driver (and this driver alone), *driver:needs* will be set 10 1
- A user running the old 10.0 version of easycam downloads the new config file.
- When he tries to install driver xyz, he will be informed that he also has to get easycam 10.1.
  When he tries to install another driver which was not affected by the bug, he does not get this message.

Newer versions of source code

The element *driver* has the attribute *driver.source*, which points back to the source code for this driver. In later versions of easycam this might be used to automatically download newer versions of the driver.

Translations

Non ASCII text (e.g. French) in source code

If non ASCII text is used in the python source code, the character encoding should be noted in the second line (under the she-bang). Otherwise the interpreter will complain after when it encounters the first accent:

```
# -*- coding: utf-8 -*-
```

Extracting the phrase from the source code

POTFILES.in lists the files that contain translatable strings.

The Build option of Glade creates POTFILES.in (in ./po), if it is not present, at the same time it creates the C-source code (in ./src).

When POTFILES.in is created, it contains the names of the C-source files, Glade has just generated. If the file already exists, it will not be touched.
If any additional files contain translatable strings, these files must be listed here.

Because the base version of easycam is written in French, the directive

```
[encoding: UTF-8]
```

has to be inserted before the file names.

The PO template (.pot) file is created using:

```
cd po
intltool-update -p
```

This creates easycam.pot (PO template).

Creating the files containing the translation

The files to be translated are named LANGUAGECODE.po (e.g. **de.po** for German, or **en.po** for english).

If the appropriate .po file does not exist, create a copy of the template file, e.g.:

```
cp easycam.pot de.po
```

Translate the .po file

Use gtranslator to translate the .po files. gtranslator can also compile the .po file to the binary format. If you press the “Compile” button, it produces: easycam.gmo

You can compile the file manually using:

```
msgfmt -o easycam.mo de.po
```

Move easycam.mo to the appropriate locale subdirectory. If you have used gtranslator, the .gmo file has also to be renamed into .mo.

Updating an existing translation

A new release of the source code will most likely contain additional phrases to translate:

First, create the template file in the usual way:

```
intltool-update -p
```

This will create a new .pot file with all translatable phrases (old and new).

To update the old po-files with the new phrases use:

```
intltool-update -d de
intltool-update -d en
...
```

The new file will contain already translated phrases, but also the new ones.

The locale subdirectory – Where to store the .mo files

easycam uses *./locale* as locale subdirectory. This means that the compiled language files (easycam.**mo**) have to be copied to:

easycam/locale/`LANGUAGECODE`/LC_MESSAGES/

I.e. the path for the German translation is:

easycam/locale/**de**/LC_MESSAGES/easycam.mo

For the English translation:

easycam/locale/**en**/LC_MESSAGES/easycam.mo

Using the translated phrases

In the GUI created by Glade

You can use the translations for messages in the GUI (stored in easycam.glade), as well as in messages in the source code.

To activate the translations when using the Glade file use the following code:

```
gtk.glade.bindtextdomain("easycam","./locale")
gtk.glade.textdomain("easycam")
```

In normal python source code

If you also need the translations n the python source code, add the following code:

```
gettext.bindtextdomain("easycam","./locale")
gettext.textdomain("easycam")
_ = gettext.gettext
```

Translatable strings in the python source code look like this:

```
print _(“Quitter”)
```

will print “Exit” in an English environment.

The “_” macro serves two purposes:

1. When executed, it calls the translation routine.
2. It marks this string as 'translatable' for *intltools-update* to find it.
   (Don't forget to put the name of the python file into POTFILES.in, otherwise the python source code will not be searched).

Hints

Finding USB IDs in driver source code

Each USB driver has a table containing the USB IDs of the device it is able to control. The table looks like this:

```
static struct usb_device_id pwc_device_table [] = {
     { USB_DEVICE(0x0471, 0x0302) }, /* Philips models */
     { USB_DEVICE(0x0471, 0x0303) },
   ...
     { USB_DEVICE(0x0d81, 0x1900) },
     { }
};
```

Searching for **struct usb_device_id** is a good way to find it.