OPP – Software Translation

Denis SAUNIER, Geoffrey BERGE,
Thibaud LAMARCHE, Thomas BERTHOME

June 4, 2014

Referring professor : Agnès ARNOULD
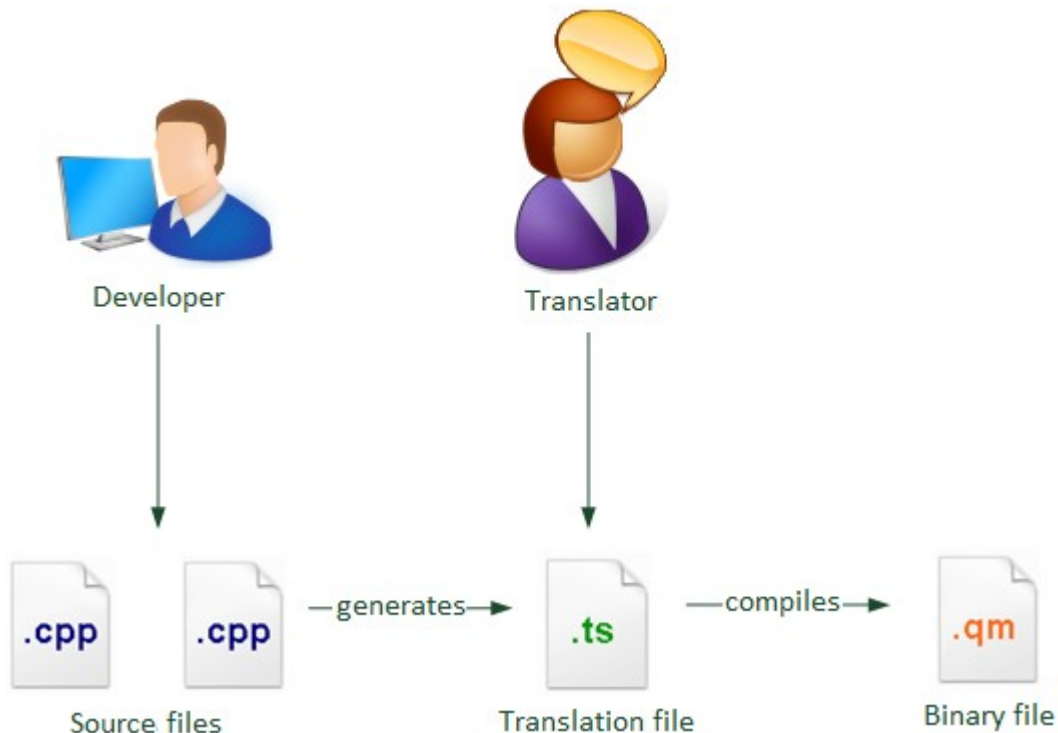
catalogue ouvert du cinéma

Summary

Denis Saunier, Geoffrey Bergé, Thomas Berthomé, Thibaud Lamarche

Software translation

Development

The first step of the translation is to write the code appropriately so that the translator can retrieve all messages to be translated.

Use QString

Qt uses its own class QString to manipulate strings. This class supports Unicode so all types of characters are allowed. Strings that must be translated in the software program should be handled with a QString.

```cpp
QString s = "MyString";
```

The tr() method

The tr() method indicates that the string can be translated.

```cpp
QMessageBox::warning(this,QString(tr("CanBeTranslated")),
        Qstring("CanNotTranslated"));
```

You can add a message to explain the context to the translator.

```cpp
saveAction->setShortcut(QKeySequence(tr("Ctrl+S", "Shortcut for save")));
```

Denis Saunier, Geoffrey Bergé, Thomas Berthomé, Thibaud Lamarche

File translation (.ts)

The software program already generates opp_fr.ts file for the French translation because in the opp.pro file, there is the following code:

```
TRANSLATIONS = opp_fr.ts
```

To generate a new translation file, it must be added at the end of the line of code. Example to add a Spanish translation file:

```
TRANSLATIONS = opp_fr.ts opp_es.ts
```

In order to generate the translation file you can use QtCreator or you can use the console.

- In a console, go to the folder of your project.

    Enter:

```
lupdate opp.pro
```

- With QtCreator:

    Tools → External → Linguist → Update Translations (lupdate)

A message of this type is obtained:

```
Updating 'opp_XX.ts'...
       Found XXX source text(s) (XXX new and XXX already existing)
```

It will update only the strings that have changed. The generated file is in the project folder.
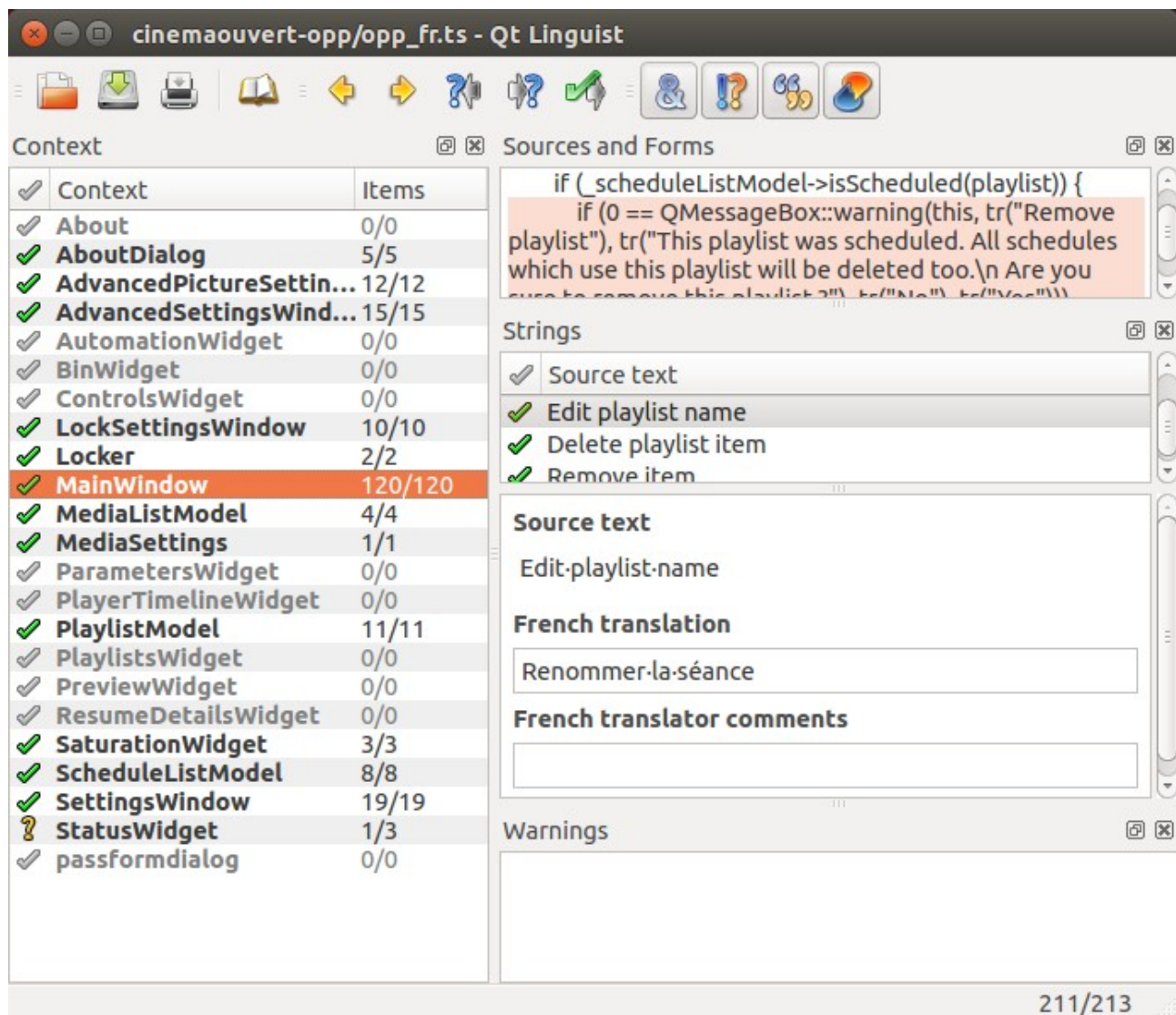
Translation with QtLinguist

To begin the translation you must open the opp_XX.ts file with QtLinguist.

Using Ubuntu or Debian, if QtLinguist is not installed, you must install it via a terminal qt4-dev-tools.

    Enter:

```
sudo apt-get install  qt4-dev-tools
```

Denis Saunier, Geoffrey Bergé, Thomas Berthomé, Thibaud Lamarche

This is the interface of Qt linguist.

You can see that the interface is separated in several windows:

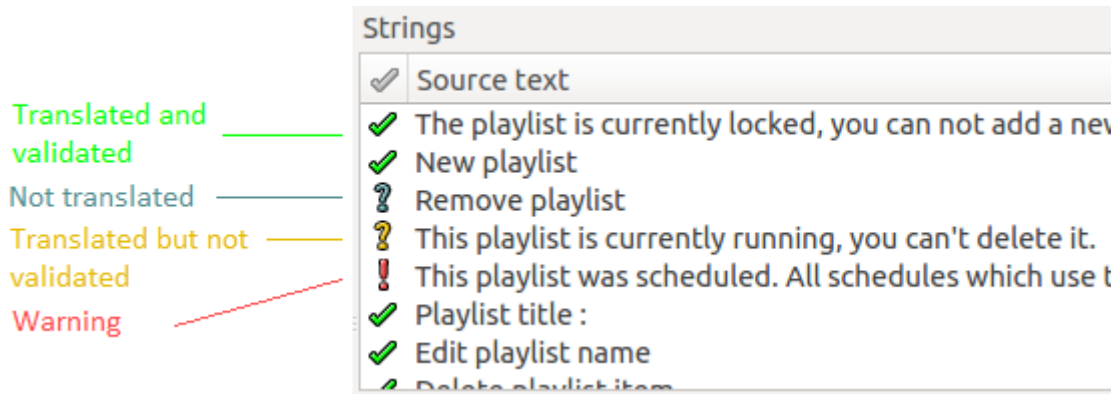Context: list of source files containing translatable strings.

Sources and forms: contents of the file selected.

Strings: list of translatable strings of the file selected.

Under "Strings": there is the English version of the string, must give the translated version.

Warnings: displays useful warnings such as "Translation does not end with the same punctuation as the source text".

Denis Saunier, Geoffrey Bergé, Thomas Berthomé, Thibaud Lamarche

Each message can have 4 states:

To validate the translation, you must click on the "❓ ".

Binary file (.qm)

You should now compile the translated file (.ts) into binary file (.qm).
To generate the binary file you can use QtCreator or you can use the console.

- In a console, go to the folder of your project.
  Enter:
  > To compile the selected file:

```
lrelease  opp_XX.ts
```
  > To compile all the project files:

```
lrelease opp.pro
```
- With QtCreator:
  > Tools → External → Linguist → Release Translations (lrelease)

A message of this type is obtained:

```
Updating 'opp_XX.qm'...
        Generated XXX translation(s) (XXX finished and XXX unfinished)
```

You should know that lrelease compiles only strings marked as completed (with the green symbol in QtLinguist).

The generated file is in the project folder.

                Denis Saunier, Geoffrey Bergé, Thomas Berthomé, Thibaud Lamarche

Load the file in the application

The file is loaded in the main() in the main.cpp file.

```cpp
int main(int argc, char *argv[])
{

        QApplication a(argc, argv);

        ...

        /*Translation file*/
        QString translationFile = "opp_";
        translationFile+=settings.value("lang").toString();

        QTranslator translator;
        translator.load(translationFile);
        a.installTranslator(&translator);

        ...
}
```

The code "settings.value("lang").toString()" returns the value corresponding to the language selected in the preferences. This value corresponds to the two letters present in the traduction file name (opp_XX.ts). These values are defined in the method "QString SettingsWindow::getLang(int index)'' of the settingswindow.cpp file.

```cpp
QString SettingsWindow::getLang(int index)
{
        switch(index)
        {
               case 0:  //English
                      return "en";
                      break;
               case 1:  //Français
                      return "fr";
                      break;
               default:
                      return "en";
        }
}
```

To add a language, you must add, in the last position, the choice of the language in the QComboBox named "comboBox_language" in settingswindow.ui file and add a "case" in the "switch case" in the previous method that returns the two-letter language.

                    Denis Saunier, Geoffrey Bergé, Thomas Berthomé, Thibaud Lamarche