Using FFMPEG in C++ with Qt Creator

Angelo Antonio Salatino[1]

Last revision: 28th Jan. 2013

0.9. Abstract

This paper aims to give some tips that allows programmers to use FFMPEG library in C++ and to create an own project with Qt Creator. First of all, I want to say sorry about my english, but this is the correct way to improve it and make more international this wiki. The examples that will be shown below are completely tested on Ubuntu OS.

Keywords: FFMPEG with C++; Qt Creator

1. Introduction

Nowadays, there are a lot of tools that simplifies the programmer's life, sometimes this kind of tools are not compatible together. Two of them are Qt Creator and the FFMPEG library. The first one is a cross-platform IDE that gives the opportunity to create C++ project with GUI but the second one is a complete, cross-platform solution to record, convert and stream audio and video. FFMPEG has been distributed in two ways, a built software (retrievable launching "*sudo apt-get install ffmpeg*" on linux systems) and a library format that should be compiled to be used (http://ffmpeg.org/trac/ffmpeg/wiki/CompilationGuide).
Despite these beautiful words, sometimes is impossible to include the FFMPEG library inside an own C++ project. In the sections below will be shown how it is possible.
To make an own project with that library it is possibile to use both the two ways, but the first one is not a good way to proceed, because after downloading it, there is only a built software that gives less benefits than the library. On the other hand, compiling the library, it gives the complete access to the function, structures, that make it a better project. The paper is organized as follow: Section 2 describes how to launch the built software as a system call and Section 3 describes how to use FFMPEG library inside C++ code.

2. FFMPEG as a system call

The built version of FFMPEG can be called by shell with an argument list (see the wiki on the official web site), so to use this into a C++ code it is necessary to implement a QProcess that can start a program.
Taking a simple example that is able to extract and save the audio stream from a input video file, here are the shell call:

[1] At the moment is a student of Computer System Engineering at Polytechnic of Bari (BA) Italy. *E-mail address* a.salatino@studenti.poliba.it.

```
ffmpeg -i inputMovie.avi -acodec pcm_s16le outputAudioFile.wav
```

Inside Qt Creator it is necessary to write these instructions:

```
QProcess _FFMPEG;
QString _process = "ffmpeg";
QStringList _paramList;
        _paramList << "-i"
                   << "inputMovie.avi"
                   << "-acodec"
                   << "pcm_s16le"
                   << "outputAudioFile.wav";

_FFMPEG.start(_process, _paramList);

if ( !(_FFMPEG.waitForFinished()) )
    qDebug() << "Conversion failed:" << _FFMPEG.errorString();
else
    qDebug() << "Conversion output:" << _FFMPEG.readAll();
```

The _FFMPEG is an object that can start an external program and communicate with him. To the *start* function is necessary to give the name of the process (ffmpeg) and the argument list (_paramList).

3. FFMPEG inside C++

As it was said above, to use the library function inside a own project is necessary to compile the library (http://ffmpeg.org/trac/ffmpeg/wiki/CompilationGuide). The FFMPEG library is completely written in C99, so sometimes is impossible to call some functions into C++ code, because the compiler gives an undefined reference error when a function from library is called. To walk around this problem, is necessary to write a wrapper that allows to call that functions. This wrapper is a header file that contains all the call to the FFMPEG functions. After that into the C++ code, the wrapper should be called with an *extern* keyword. Here are some code:

wrapper.h

```
#ifndef WRAPPER_H_
#define WRAPPER_H_

#include <libavutil/opt.h>
#include <libavcodec/avcodec.h>
//other FFMPEG inclusion

// Your function and structure …
```

```
    void service (... , ...){ //a generic function
            ...
     }

    #endif
```

main.cpp

```
    #include <QtGui/QApplication>
    #include "mainwindow.h"

    extern "C"{       //here the call to the wrapper with extern keyword
      #include "wrapper.h"
    }

    int main(int argc, char *argv[])
    {
              ….

              service(... , ...);     //calling a function into the wrapper
              QApplication a(argc, argv);
              MainWindow w;
              w.show();
              return a.exec();
    }
```

fooProject.pro

```
            HEADERS += wrapper.h

    QMAKE_CXXFLAGS += -D__STDC_CONSTANT_MACROS

    LIBS += -pthread
    LIBS += -L/usr/local/lib
    LIBS += -lavdevice
    LIBS += -lavfilter
    LIBS += -lpostproc
    LIBS += -lavformat
    LIBS += -lavcodec
    LIBS += -ldl
```

```
        LIBS += -lXfixes
        LIBS += -lXext
        LIBS += -lX11
        LIBS += -lasound
        LIBS += -lSDL
        LIBS += -lx264
        LIBS += -lvpx
        LIBS += -lvorbisenc
        LIBS += -lvorbis
        LIBS += -ltheoraenc
        LIBS += -ltheoradec
        LIBS += -logg
        LIBS += -lopencore-amrwb
        LIBS += -lopencore-amrnb
        LIBS += -lmp3lame
        LIBS += -lfaac
        LIBS += -lz
        LIBS += -lrt
        LIBS += -lswresample
        LIBS += -lswscale
        LIBS += -lavutil
        LIBS += -lm
```

In conclusion with this simple trick we are able to use the power of the FFMPEG library inside in a own project.
I would to say thank you to whoever want to improve this paper.