Guide to DVD Chapter 22 Examples: *Andrés Cabrera*

Graphical User Interfaces for Audio Programs Using the Qt Toolkit

The Qt Toolkit

The Qt toolkit is a cross-platform SDK which allows programmers to build cross-platform programs very easily. In practice, when the Qt toolkit is used throughout a program -avoiding platform specific functions or libraries-, it can be built without changes from the same sources on all platforms supported by Qt, which incude Windows, OS X and Linux among others. The Qt toolkit is free software and has been licensed as LGPL before version 4.5 and as GPL for later versions. It can be downloaded from: http://www.qtsoftware.com/

You can download binary or source packages, but building Qt takes quite a long time, so getting a binary package is highly recommended.

The Qt Build System

Qt uses its own build configuring system called qmake. The qmake program reads the information from .pro files and creates a Makefile, XCode or Visual Studio project to build the program and link to Qt as necessary. The actual name of the .pro file is not important as Qt will look for any file with the extension .pro in the current directory. So to build a Qt program it is usually necessary to do:

```
$ qmake
```

On Windows and Linux you can then build using make or nmake depending on your compiler. On OS X, an XCode project is generated.You can also generate Visual Studio projects on Windows by doing:

```
$ qmake -t vcapp -o offlinerender.dsp offlinerender.pro
```

If you are on OS X or you have asked for a VS project, just open the project and build from there.

Building the Examples

Building the examples is easy, but it is necessary to have libsndfile and portaudio installed. There are four examples included: offlinerender, offlinerender2, realtimerender, realtimerender2. These files show how to build Qt interfaces to audio programs both for file processing (offline) and for realtime processing. To build the examples, navigate to the example folder you want (e.g. The folder `realtimerender/`), and there do:

```
$ qmake

$ make
```

This will generate the executable. Remember to open the XCode project on OS X instead of running make. On Windows, you will need to open the VS project or use nmake depending on your development environment.

The folder `lazzarini/` contain files from Victor Lazzarini's Chapter 5 “Time-domain Audio Programming”, and is not meant to build. Rather, the files here are used as the audio algorithms for the GUIs. The folder compressor/ contains files from DVD Chapter 18: “Dynamic Range Processing”.

Libraries

The offline rendering examples work on soundfiles, and require libsndfile, which can be downloaded from www.mega-nerd.com/libsndfile. See Book Appendix C for more details. The realtime examples use portaudio for I/O. See the Book Appendix D for more details.

OfflineRender Example

This example shows a simple GUI to select input and output audio files. After building it according to the instructions above, run it by double-clicking on its icon or do:

```
$ ./offlinerender
```

The upper line shown on the GUI corresponds to the input file, and the lower one to the output file. You can use the browse buttons beside the lines to select each file. When you have selected both files, press the Process button. When you do this, the input file will be processed using the flanger algorithm from book chapter 5, with a fixed delay time of 0.01 seconds and a feedback of 0.8 (80%).

OfflineRender2 Example

This example builds on the previous one, adding a progress barr and sliders for the delay time and feedback. After building it according to the instructions above, run it by double-clicking on its icon or do:

```
$ ./offlinerender2
```

The delay time slider has a range from 1 to 50 miliseconds and the feedback goes from 0% to 99%.

RealtimeRender Example

This example creates a GUI to select the audio interfaces for audio I/O. It only passes he audio from its input to the output. After building it according to the instructions above, run it by double-clicking on its icon or do:

```
$ ./realtimerender
```

Be careful with the input an output gains from the soundcard as feedback is very likely if the input is a microphone.

RealtimeRender2 Example

This example uses the previous example, and adds widgets to control in realtime the knee compressor algorithm from DVD chapter 18. After building it according to the instructions above, run it by double-clicking on its icon or do:

```
$ ./realtimerender2
```

You can control input gain and compressor parameters like threshold, ratio, attack and release times and knee. Be careful with the input gain as feedback is very likely.