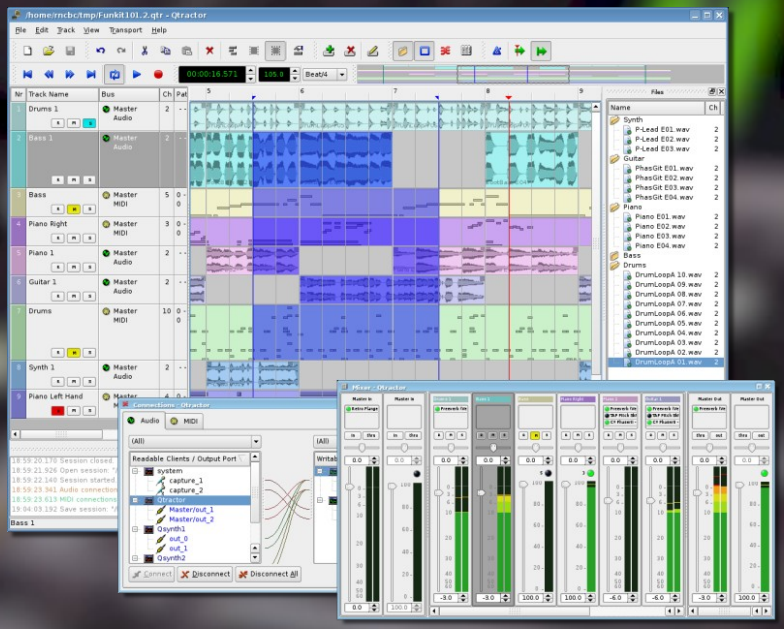


Qtractor

Linux Audio/MIDI Multi-Track Workstation

Version 0.1.1

Futile Duchess

February 2008

User Manual Revision 0.1.1

by

Rui Nuno Capela and James Laco Hines

1. Abstract

Qtractor is an multi-track Audio and MIDI sequencer application written in C++ around the Qt4 toolkit using Qt Designer. The initial target platform will be Linux, where the Jack Audio Connection Kit (JACK) for audio, and the Advanced Linux Sound Architecture (ALSA) for MIDI, are the main infrastructures to evolve as a fairly-featured Desktop Audio/MIDI Workstation GUI, specially dedicated to the personal home-studio.

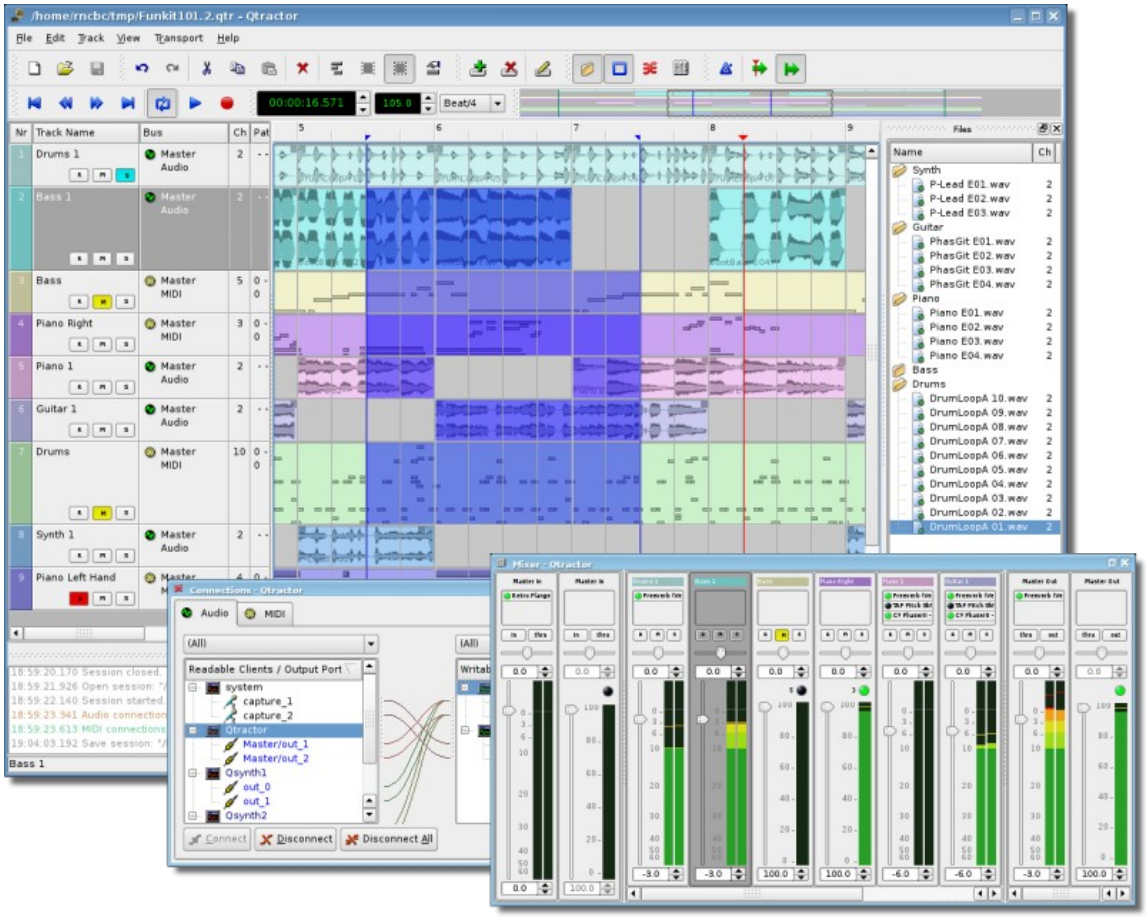


Figure 1: Main GUI Window: Tracks, Mixer and Connections

2. Introduction

As a new Linux Audio offering, Qtractor [1] targets and positions itself comfortably tagged for the *techno-boy bedroom home-studio*. However, in general, it is just yet another digital audio and MIDI multi-track composition and arranger software application. The design and functionality model takes as fundamental, the now usual multi-track composing techniques for modern music-making. It aims to be intuitive and easy to use, yet powerful enough for the serious recording enthusiast.

Qtractor is designed to be the heart of your home studio. Even though built on a monolithic application architecture (Linux kernel), this is not the whole of it. Besides its name, it should be noted, Qtractor is not a music *"tracker"* type of program. However, it has all the potential to do that job, if needed.

Qtractor is a non-destructive sequencer and arranger. It does not affect, alter or modify in any way, the audio and/or MIDI files that are displayed as Clip Objects. What *is* destructive are files resulting from capture and recording operations, and explicit changes made through specialized Clip editing (e.g., *MIDI Editor*).

Currently the Qtractor project is simply the hobby of one developer. Development was started April of 2005, initially as a Qt3 application. Since October 2006, it is officially a Qt4 [2] application.

Qtractor is natively hardwired and exclusive to the JACK [3] audio infrastructure, and the ALSA [4] sequencer for MIDI, thus currently being a Linux-only application.

Qtractor is free open-source software, licensed under the GPL and is welcoming all collaboration and review from the Linux Audio developer and user community in particular, and the public in general.

3. Linux System Requirements

The software requirements for build and run-time are listed as follows:

Mandatory:

- **Qt 4** (core, gui, xml) - C++ class library and tools for cross-platform development and internationalization.
  http://www.trolltech.org/products/qt/

- **JACK** Audio Connection Kit,
  http://jackaudio.org/

- **ALSA** - Advanced Linux Sound Architecture,
  http://www.alsa-project.org/

- **libsndfile** - C library for reading and writing files containing sampled sound,
  http://www.mega-nerd.com/libsndfile/

- **LADSPA**- Linux Audio Developer's Simple Plugin API,
  http://www.ladspa.org/

Optional Support Libraries (opted-in at build time):

- **libvorbis** (enc, file) - Ogg Vorbis audio compression,
  http://xiph.org/vorbis/

- **libmad** - High-quality MPEG audio decoder,
  http://www.underbit.com/products/mad/

- **libsamplerate** – The secret rabbit code, C library for audio sample rate conversion,
  http://www.mega-nerd.com/SRC/

- **liblo** – Lightweight OSC implementation (needed for DSSI GUI support)
  http://liblo.sourceforge.net/

- **DSSI** – An API for soft synth plugins with custom user interfaces
  http://dssi.sourceforge.net/

- **VST-SDK** – Steinberg's Virtual Studio Technology
  http://www.steinberg.net/

4. Download

Qtractor is still in its alpha stages of development, but already fully functional. The latest versions are publicly available from the qtractor.sourceforge.net project web site [1].

The bleeding-edge source code may be found in the CVS repository, through anonymous (pserver) access with the following instructions:

Login to the CVS repository:

```
cvs -d:pserver:anonymous@qtractor.cvs.sourceforge.net:/cvsroot/qtractor login
```

When prompted for a password, hit enter and proceed for check-out:

```
cvs -z3 -d:pserver:anonymous@qtractor.cvs.sourceforge.net:/cvsroot/qtractor co qtractor
```

Prepare the `configure` script on the just created `qtractor` source tree directory:

```
cd qtractor
make -f Makefile.cvs
```

Hopefully, the source tree will be now ready for build and installation.

5. Linux Installation

The standard procedure for source distributions based on *autoconf* follows, through the now quite usual build command sequence:

```
./configure && make
```

and complete installation by having the appropriate (root) authority:

```
make install
```

which will end installing by copying the `qtractor` binary executable and desktop and icon files to common standard base system locations.

NOTE: To see all configuration options of the configure command, use ./configure --help

6. Configuration

Qtractor holds its run-time settings and configuration state per user, in a file located as `$HOME/.config/rncbc.org/Qtractor.conf` . Normally, there is no need to edit this file, as it is recreated and rewritten every time `qtractor` is run.

7. Master GUI (Main)

The Qtractor user interface is a design that is modern and standard for most audio workstations of the present. The interface is easy for average user interaction, and intuitive enough that you can easily interact and discover the software's full potential of the inner-core application.

Figure 1 shows an overall aspect of the GUI, with an example session loaded into the workspace.

The main Qtactor window is initially laid out in this fashion:

- Menu and Tool-Bars at the top
- Track list information on the left
- Track (audio region) data on the right, with time scale above it.

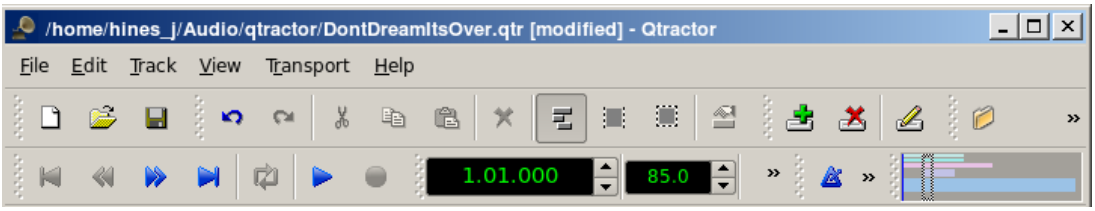


Figure 2: The Main Menu and Toolbar

The Track Data section is where most of the action takes place, and contains visual displays of the waveforms, or audio data This section is used for editing Clip objects and maneuvering around your Project Session.

Qtractor also has other useful windows, such as the Mixer window, and the Connections window. Both these windows can be opened using the F8 key for Connections window (Jack PatchBay), and the F9 key for the Mixer window. These items are also selectable from the View menu.

Two utility windows are additionally featured: the *Messages* window, specially suited for debugging, and the *Files* window, where audio and MIDI files are organized and selected on demand.

Dialog windows for editing *session*, *track* and *clip* properties are also accessible in their proper context, which will be discussed in their respective sections.

Finally, session and application configuration options are assisted through respective customizing dialogs: *Buses, Instruments and Options.*

8. Sessions

8.1 Understanding the Session Project

A Qtractor Session Project contains all the information about all your Clip Objects, placement of Clip Objects, Mixer setup, Plugins, Tempo, Time Signature and Connections Patchbay. When creating or saving a project, all this, and any related settings are saved on your hard disk within this Session Project file.

It is important to note that Qtractor Sessions are locked to a Session Project sample rate. This is dependent on the sample rate of the JACK[3] server running at the time the session is created.

Any attempt to convert non-matching sample-rate sessions will result in a recommendation warning message.

However, individual audio clip files are automatically converted on playback in real-time to the host sample-rate (via libsamplerate [8]). This method, while it works very well, is not the recommended method due to possible errors in the real-time sample rate conversion.Realtime sample rate conversion is also going to use quite a bit more valuable CPU resources.

Please note: ***It is NOT recommended you record audio tracks while in this sample rate conversion mode.***

Rui Nuno Capela is working toward eliminating this shortcoming by taking control of JACK from within Qtractor, and restarting it using the Session's project parameters. This will ultimately reconnect any plugins, set the proper sample rate, etc. Until this feature is available, please follow the recommendations listed above.

8.2 Time Signature and Tempo

Although you may select any time signature and tempo for your Session, by present design Sessions can only contain a single constant tempo. Tempo must be regarded as a global setting of a Session. Qtractor does not presently support Tempo Mapping, but may eventually do so.

A new feature, which applies to existing audio clips, will reflect all tempo changes with a corresponding time-stretching effect. Time-stretching is thus applied in real-time at the buffering level, as a custom WSOLA algorithm based and derived from the SoundTouch [12] library.

8.3 Session Properties

To access the Session Properties, access the File/Properties menu item. Here, you can name your Session Project, set the Time Signature, Bars/Beats, Ticks/Beat, etc.

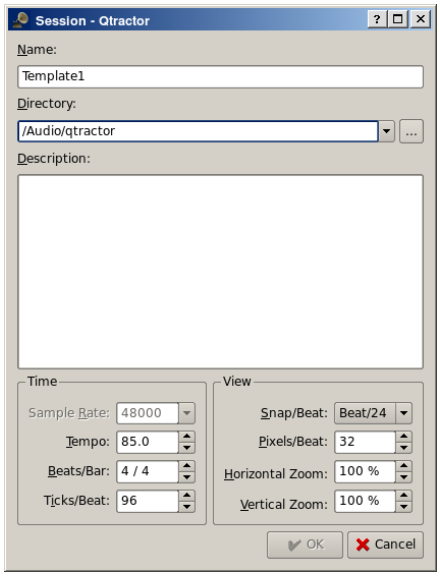


Figure 3: Session Properties Dialog

8.4 Session Options

Access the Options windows via the View/Options menu item. This window, and its tabs allows

you to control the global parameters of Qtractor, and these settings are saved within your Session file.

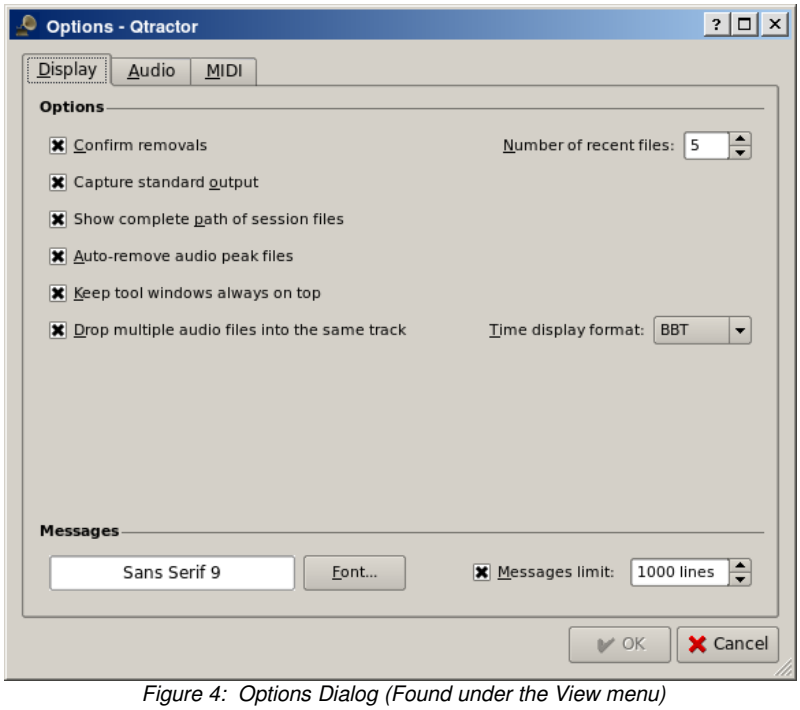
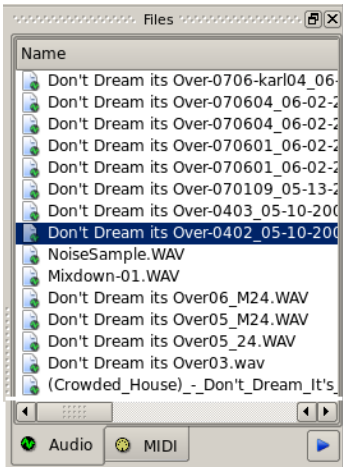


Figure 4: Options Dialog (Found under the View menu)

9. Files

Sound file selection is made available through a tabbed mini-organizer. Audio and MIDI file lists are kept separate on their respective tabs. Files can be explicitly added and grouped into a hierarchical tree list. Individual and multiple files can be drag-and-dropped from the desktop environment and within the provided tree list. This lists all the files which are referred in the working arrangement session. File items can be drag-and-dropped directly into the track window, thus creating new clips in the working arrangement. This is mainly used as a your audio/MIDI file data pool.



The Files Pool can also allow you to preview the files shown in the windows either by double-clicking the file name, or by click the play button on the lower right hand side of the Files Pool window.

Audio file format support is provided by libsndfile [5], and supports wav, aiff, flac, au, etc. Optional libraries provide support for both ogg and mp3 formats. libvorbis [6] (ogg) and libmad [7] (mp3). MIDI file support covers the usual SMF formats 0 and 1, through a native, home-brew implementation.

10. Clip Objects

10.1 Clip Object Summary

Clip Objects are the elemental items of a session arrangement, and can contain either Audio or MIDI data. A Clip Object is merely a region of an actual sample or MIDI file. A Clip Object is also non destructive, and can be copied, truncated and time stretched as if it were actual audio/midi data.

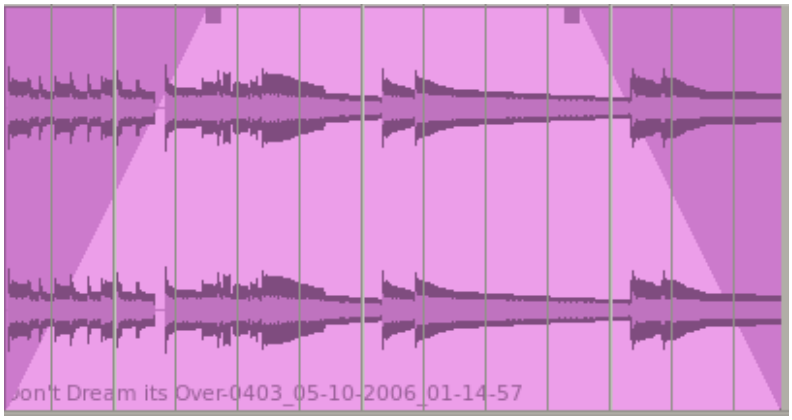
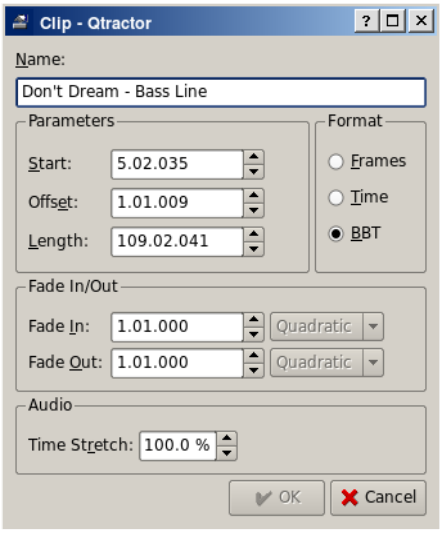


Figure 6: An example of an audio Clip with Fade-In and Fade-Out applied.

10.2 Clip Properties (Audio)



Clip properties include its label (name), start time (location), offset and length (in frames), fade-in and fade-out length (in frames) and Tim Stretch percentage, respectively, from the start and end of the clip. Although fade-in and fade-outs are always displayed as straight lines, the actual audio volume (gain) and MIDI velocity effect can be opted to be of either linear, square or cubic characteristic, in as for an approximation to the logarithmic model of human ear perception.

Figure 7: Clip Properties

Clips are placed on tracks, either by importing audio and MIDI files as new tracks, or by dragging and dropping files into the track-view arranger window. After being placed on their respective tracks, you may perform clip-region operations such as drag, copy, cut, paste, delete, truncate, fade in/out etc. Altering clip fade-in and fade-out is accomplished by dragging the handles (square boxes) on the top ends of any clip.

Most clip editing operations are accomplished through the usual mouse interaction, by first selecting one or multiple clips and/or regions, and applying the edit action upon the resulting selection. There are three selection modes available: clip, range and rectangular modes.

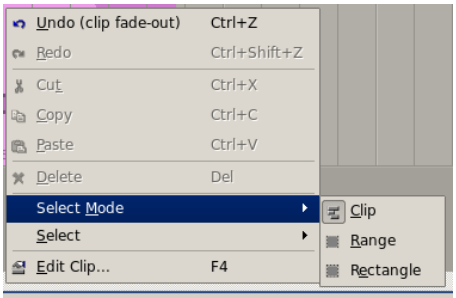


Figure 8: Demonstrates a right-click over a clip allowing for Mouse Mode Selection.

- In Clip mode, Clip Objects are selected as a whole with no sub-clip regions possible.
- In Range mode, Clip Object regions are selected on all tracks between a given time interval or range.
- In Rectangular mode, only the regions that fall under a rectangular area are selected, this means for adjacent tracks and clips only.

10.3 Clips and Tracks

Tracks may be armed for recording, making way for creating new audio and MIDI clip files with captured material. Tracks can also be muted and soloed on mix-down, which also applies when exporting. Most editing operations should be possible while playback is rolling (but not completely safe though; there are many procedural helpers, but not completely assisted with lock-free primitives, yet).

Figure 9: Track Functions: Record, Mute and Solo

MIDI Clip Objects are representations of a sequence of events of one single MIDI channel, as extracted from a SMF format 0 file or of one single track, as from a SMF format 1, either in whole or in part.

11. Engines and Buses

Qtractor is a fairly massive multi-threaded application. For instance, each audio clip has a dedicated disk I/O executive thread, which synchronizes with the master engine and, for all purposes, to central JACK real-time audio processing cycle, through a lock-free ring-buffer. These audio file ring-buffers are recycled (filled/emptied) at one second threshold, and has a maximum streaming capacity of 4-5 seconds of audio sample data. Smaller clips are permanently cached in a RAM buffer.

Audio thread scheduling is mastered and mandated through the JACK callback API model. MIDI clip events are queued in anticipation through one MIDI output thread, which feeds a ALSA sequencer queue, synchronized on one second periods to the JACK process cycle. A single thread is responsible for listening (polling) for MIDI input, and multiplexes all incoming events through record-armed MIDI tracks. Time stamping is done through the ALSA sequencer facility.

Looping is made possible through the audio file buffering layer, right at the disk I/O thread context. The same consideration is adopted for MIDI output queuing. JACK transport support is not an option, as playback positioning is constantly kept in soft-chase fashion. Audio frame relocation is accounted from successive JACK client process cycles (i.e. buffer-period resolution).

On this particular design, JACK and ALSA sequencer ports are logically aggregated as buses with respect to the audio and MIDI signal routing paths, functioning as fundamental device interfaces. Input buses, through exposing their respective input ports, are responsible inlets on capture and recording. Output buses are the main signal outlets and are responsible as playback and, more importantly, as mix-down devices.
Buses are independently assigned to tracks. Each track is assigned to one input bus for recording, and to one output bus for playback and mix-down. The assigned output bus determines the number of channels the track supports. Clips bounded to disparate multichannel audio files, for which their number of channels do not match with proper bus/track's one, are automatically resolved on mix-down. Figure 2 shows one typical signal flow block diagram.

By default, “Master” buses are automatically created at session startup, being stereo for audio (2 channels ports, auto-connected) and single port for MIDI (16 logically addressable channels). Bus ports are accessible for arbitrary connection to and from external client applications or devices, through the *connections* window interface.

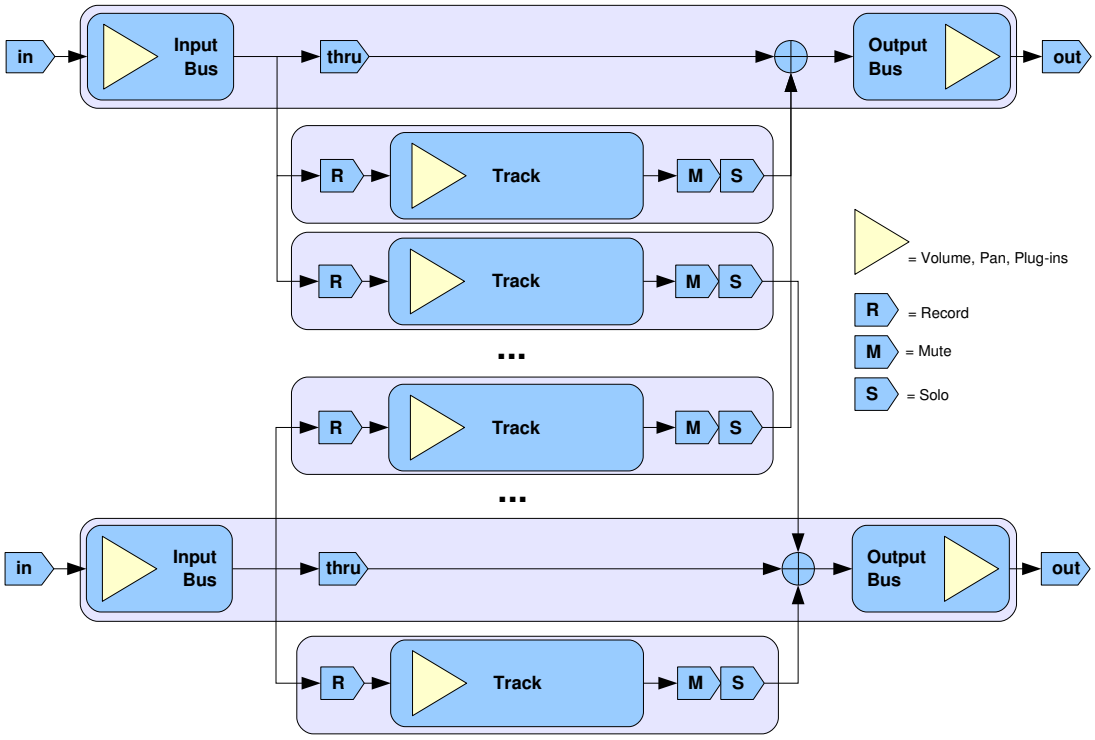


Figure 10: Buses / Tracks signal flow diagram

12. Track View

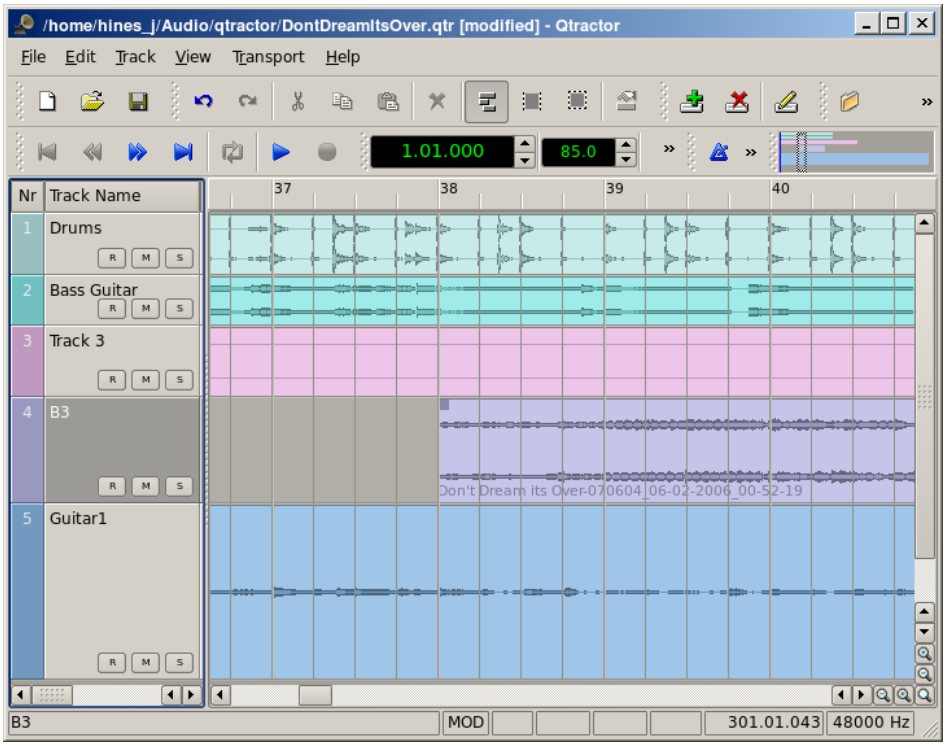


Figure 11: This is a snapshot of the main window with track layout.

Tracks are arranged as a sequence of one or more overlapping clips of the same file type, either audio or MIDI. The tracks window is the main application workspace, serving as a virtual canvas of a multi-track composition arranger. Most of the editing operations are made on this track list-view window.

The track list-view window has two panes, the left one displays the list of tracks with their respective properties and the center-right pane is the proper track-view canvas window where main multi-track composition and arranging activity is pictured and performed. As usual, tracks are stacked on horizontal strips and clips are layered on a bi-dimensional grid, in time sequence for each track strip. Time is modeled on the horizontal axis and pictured by a bar-beat scale ruler at the top of the track-view.

Clips may be conveniently aligned to discrete time positions, depending on the current *snap* mode setting. When not set to “None”, the snapping is always carried out to MIDI resolution, quantized to ticks per quarter note granularity.

Each track has its own user assignable colors for better visual identification. Audio clips are displayed with approximate waveform graphic, with peak and RMS signal envelopes as read from the respective audio file segment. MIDI clips are shown as a *piano-roll* like graphic, with note events shown as small rectangles, depicting pitch, time and duration.

All session, track and clip editing operations are undo/redo-able. Discrete view zooming and track vertical resizing operations are also available.

13. Mixer

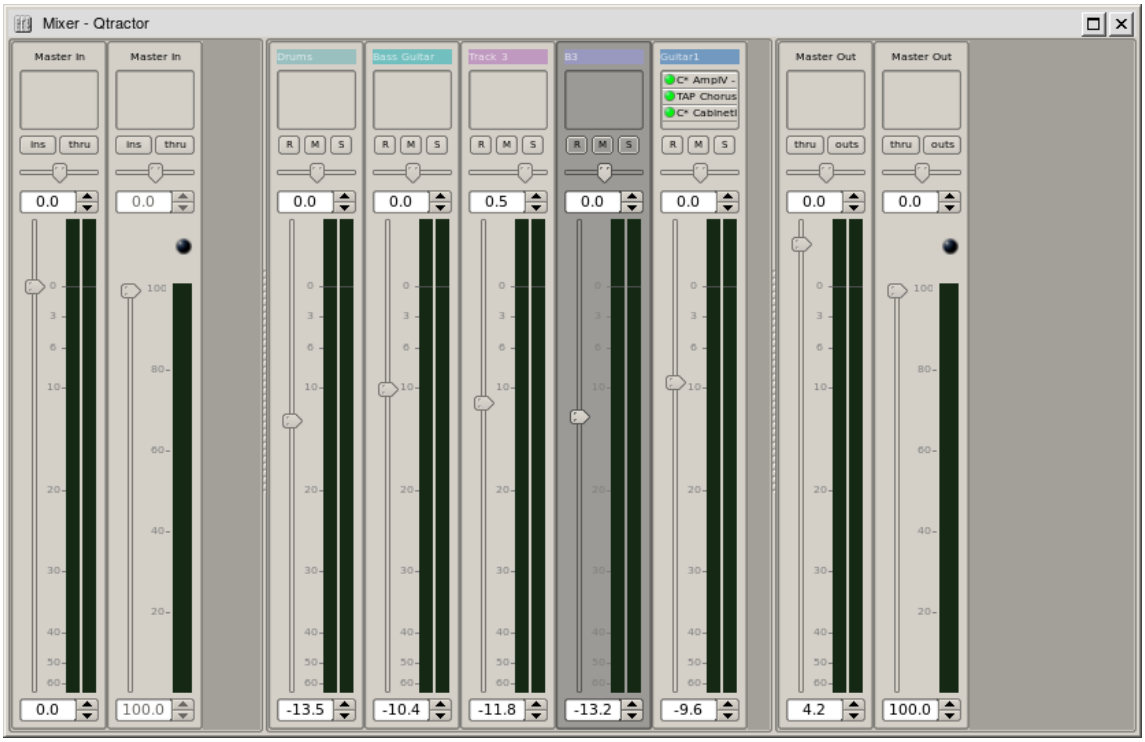


Figure 12: Full Mixer Window

The mixer window serves for session control, monitoring, recording and assistance in mix-down operations. The mixer is divided in three panes: the left accommodates all input buses, the center with individual track strips and the right for the output buses. Each mixer strip offers a volume and pan control and monitors each one of the respective buses and tracks. Audio strips also offers the possibility to chain plug-in effects (LADSPA [9]).

Monitoring is presented in the form of peak level meters for audio and note event velocity for MIDI, both with fall-off *eye-candy*. MIDI mixer strips also feature an output event activity LED.

Audio volume is presented on a dBfs scale (IEC 268-10) and pan is applied in approximated equal-power effect (trigonometric weighting). For MIDI tracks, volume control is implemented through respective channel controller-7 and system-exclusive master volume for output buses. MIDI pan control is only available for track strips and is implemented through channel controller-10. MIDI input buses have volume and pan controls disabled.

14. Connections Patch-bay

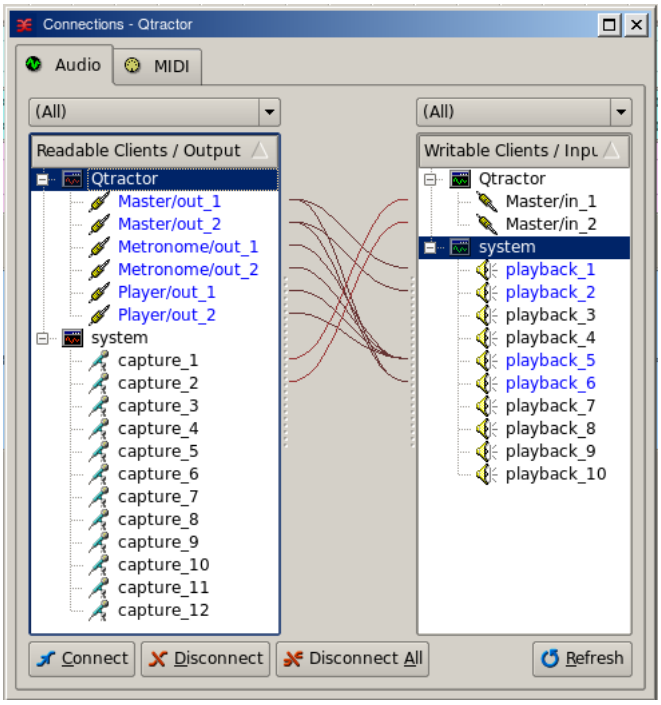
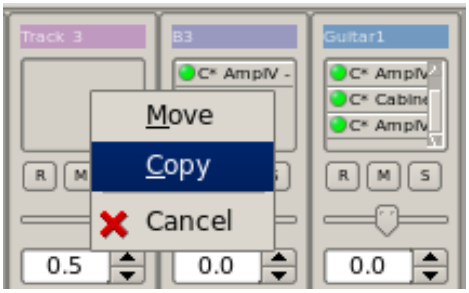


Figure 13: Jackd Connections Window

The connections window serves to establish the audio and MIDI port connections between the internal core layer input and output buses (ports), and the external devices or client applications. Incidentally the *Connections* window can also be used to make connections between external client application ports, either JACK clients for audio, or ALSA sequencer clients for MIDI. In fact, it almost completely replicates the very same functionality of QjackCtl [10]. All connections on the existing input and output buses are properly saved and restored upon session recall.

15. Audio Effects Plugins



15.1 Summary

There are three types of Plugins supported within Qtractor. LADSPA, DSSI and VST. Plugin support is available for all audio input and output buses and for all audio tracks. All Plugin types are aggregated seamlessly as one single instance on a multi-channel context and can be individually selected, activated and moved within the plug-in chain order. Also, you may drag-and-drop all plugin instances over the mixer strip channels. You can move, drag and drop inside the same strip or over to, and from any other. You may also copy plugins from one channel strip to another as well. A mini menu will ask if you want to copy or move the plugin. Individual plugin control parameters can be modified in real-time through provided dialog windows and maintained

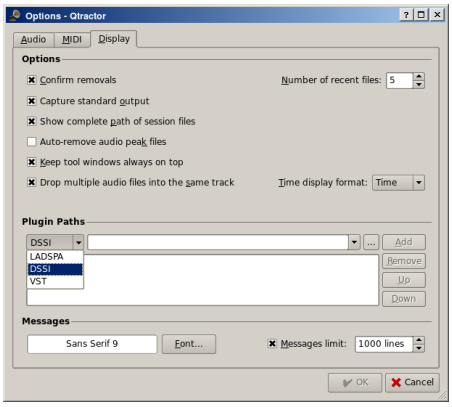
as named presets for re-usability.

15.2 LADSPA

LADSPA [9] has been the Linux audio plugin standard for many years. There are literally hundreds of LADSPA plugins available for Linux. LADSPA plugins give the user many standard options such as Eqs, Filtering, Reverb, Chorus, Amp and speaker simulation, etc.

15.3 DSSI

DSSI plugin support is available for DSSI effects plugins. DSSI Instrument plugins are not yet supported. You must have the core DSSI subsystem installed in order for this type of plugin to function. When DSSI is present, the DSSI-VST wrapper may also be used. This wrapper uses WINE (http://www.winehq.org) to allow a user to run native Windows® VST applications. The DSSI paths may be set within the Options dialog under the Display tab, or with an environment variable. (see section 15.5).



15.4 VST (Linux Native)

Native Linux VST plugin support is also available. Presently, there are only a few native Linux VSTs available, but more should be on the way soon, thanks to some aggressive ongoing projects.

NOTE: Native Linux VST support does NOT include running of Windows® VST plugins. Please use the DSSI-VST wrapper when attempting to use this type of plugin, and make sure your Windows® VST plugins are located within your DSSI path environment variabe.

15.5 Compiling Qtractor for Native Linux VST Support

VST support is not that easy. At least it does not work out of the box. First off, due to its licensing issues, you'll have to go through the nuisance and download yourself the VST SDK from the Steinberg Media Technologies GmbH site, specifically digging through the 3rd Party Developers section. It doesn't matter much whether you pick the VST 2.3 or VST 2.4 version, but you need to pick one and just one only. Do not use the recently announced VST 3.0. It will not work.

Once you have downloaded the VST SDK zip-archive, for which you'll have to accept their license and supply some personal data, you'll have to unpack the pertinent header files, which are found under the respective folder. Just copy those couple of files to somewhere on your system:

- **VST SDK 2.3**: `vstsdk2.3/source/common/`
  `aeffectx.h`
  `AEffect.h`

- **VST SDK 2.4**: `vstsdk2.4/pluginterfaces/vst2.x/`
  `aeffectx.h`
  `aeffect.h`

If you choose to copy those files into some standard include directory (eg. `/usr/local/include` or `/usr/include`) all will be handled automagically by the `./configure` build step. Otherwise you'll need do supply the path yourself, as in `./configure –with-vst=/path/to/vstsdk2.x/include`.

Once properly built, you will need to grab some native VST plugins and tell where those might be found. Currently, you'll need to set the `VST_PATH` environment variable with the paths where the plug-ins will reside and can be picked up by Qtractor.

Bash Command: `export VST_PATH=`*/path/to/vst_plugins*

Some ready made Linux VST plug-ins can be found on the following web sites:
http://www.linux-vst.com/
http://www.linux-vst.com/
http://cern.linux.vst.googlepages.com/home

18. MIDI Instruments

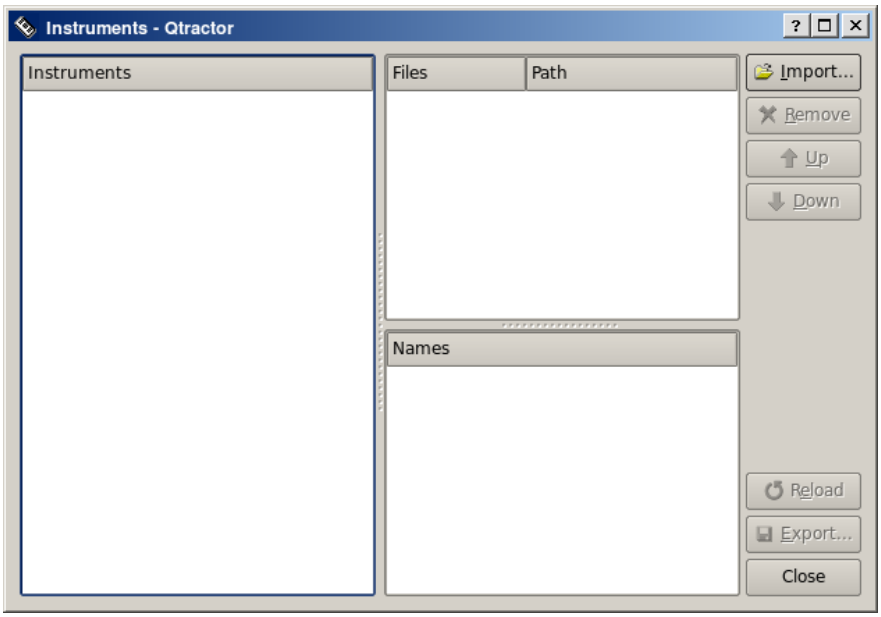


Figure 17: MIDI Instruments Editor

As a special feature, Cakewalk [11] instrument definition files (.ins) are natively supported, thus offering a convenient MIDI bank-select/program-change mapping for existing MIDI instrument patch names, and easier, intelligible selection of MIDI track channels.

19. MIDI Editor

Each MIDI clip content may be readily edited under a dedicated and fairly complete piano roll *MIDI Editor*t, with individual pitch, velocity and controller, editable trough the usual GUI operations such as: Multi-extended selection, drag-and-drop, move, cut, copy, paste, deletion of every event in the MIDI sequence is rightly accessible on the fly (Figure X).

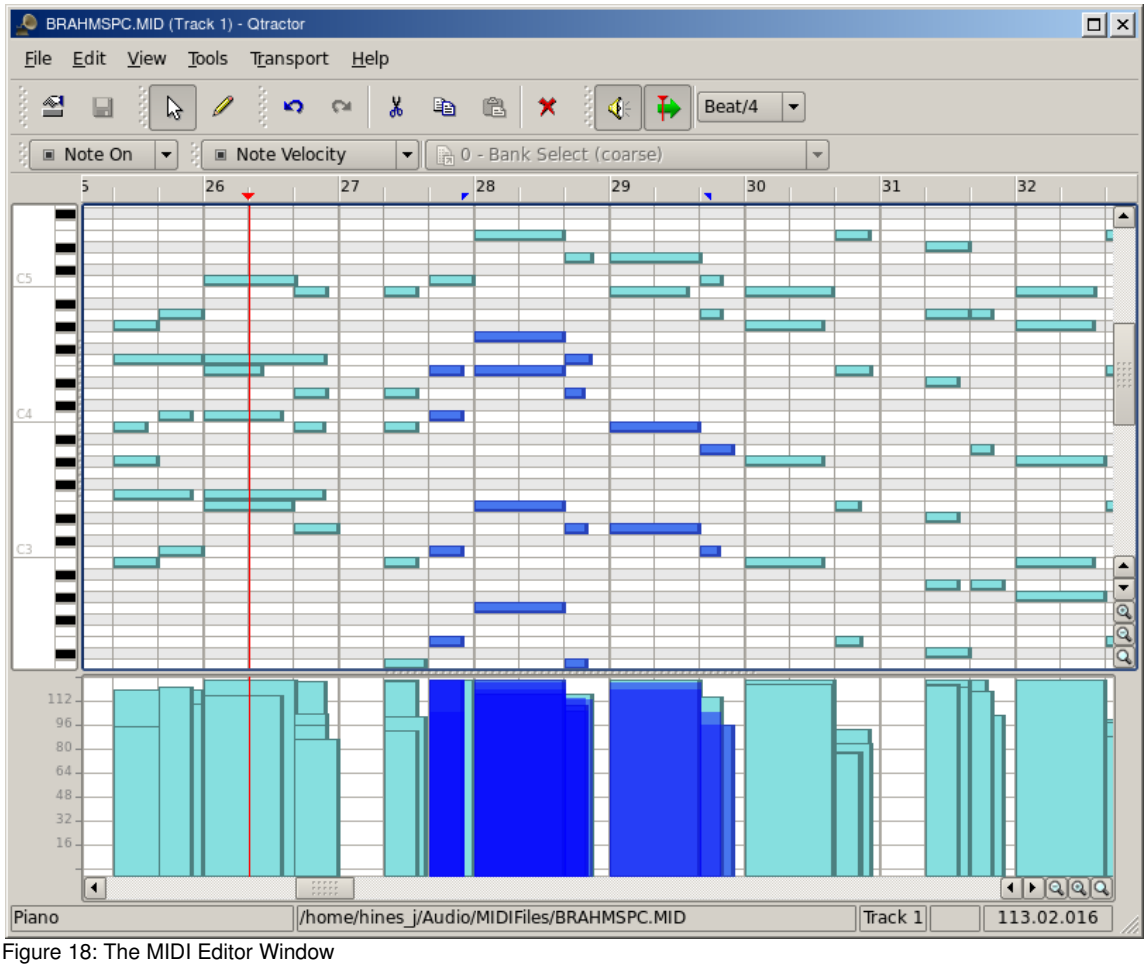
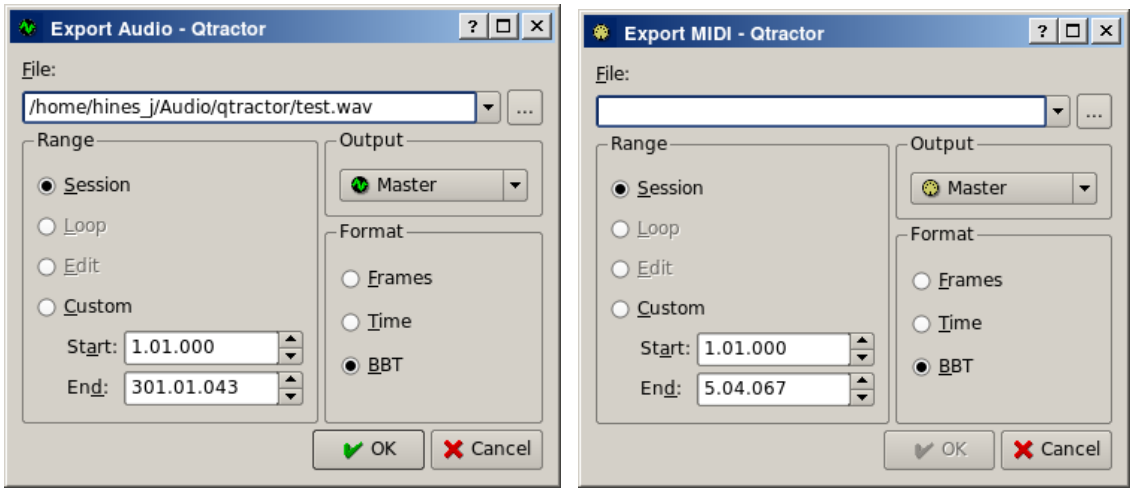


Figure 18: The MIDI Editor Window

Special home-brew tools for batch processing are also implemented and applicable to the any event selection: quantize, transpose, normalize, randomize and resize. All MIDI editing operations are available and processed in real-time, effective while playback. Several *MIDI Editor* instances may be active and open in any time, provided each one refers to its own clip.

All MIDI content may be saved as standard MIDI files (SMF Format 0 or 1).

20. Audio / MIDI Export



Figures 18 & 19: The Export Audio and Export MIDI Windows

All or part of the session may be exported to one audio or MIDI file. *Audio export* is implemented through the special JACK *freewheel* mode, thus faster than real-time, resulting in the complete and exact mix-down of selected audio material into a designated audio file of the opted format (wav, flac, au, aiff or ogg). *MIDI export* is just the same but for MIDI material only, resulting in the merging and concatenation of selected MIDI tracks and clips into a single MIDI file (SMF Format 0 or 1).

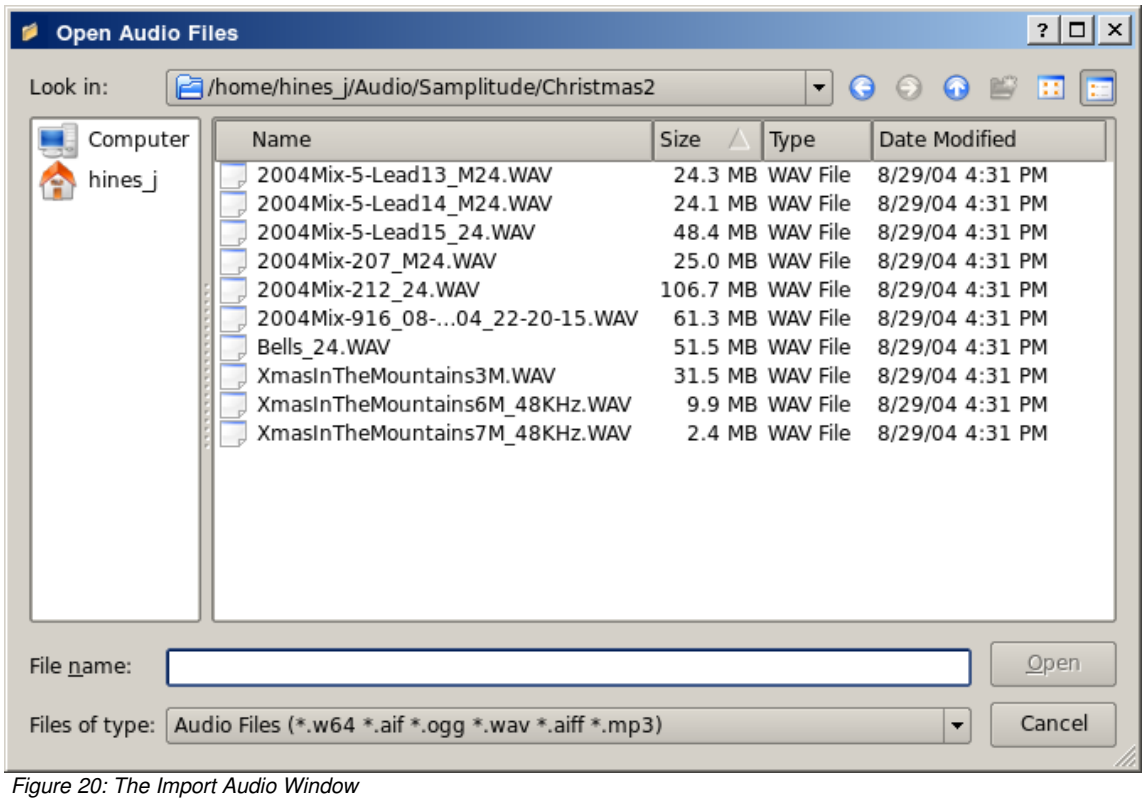


Figure 20: The Import Audio Window

21. Keyboard Shortcuts Editor

Keyboard shortcuts are useful for the power user, in such that it provides for a quick mechanism for performing often used commands quickly, without the use of your mouse.

Keyboard shortcuts may be customized to your preference by using the Shortcuts Editor. This editor may be found in the Help menu.

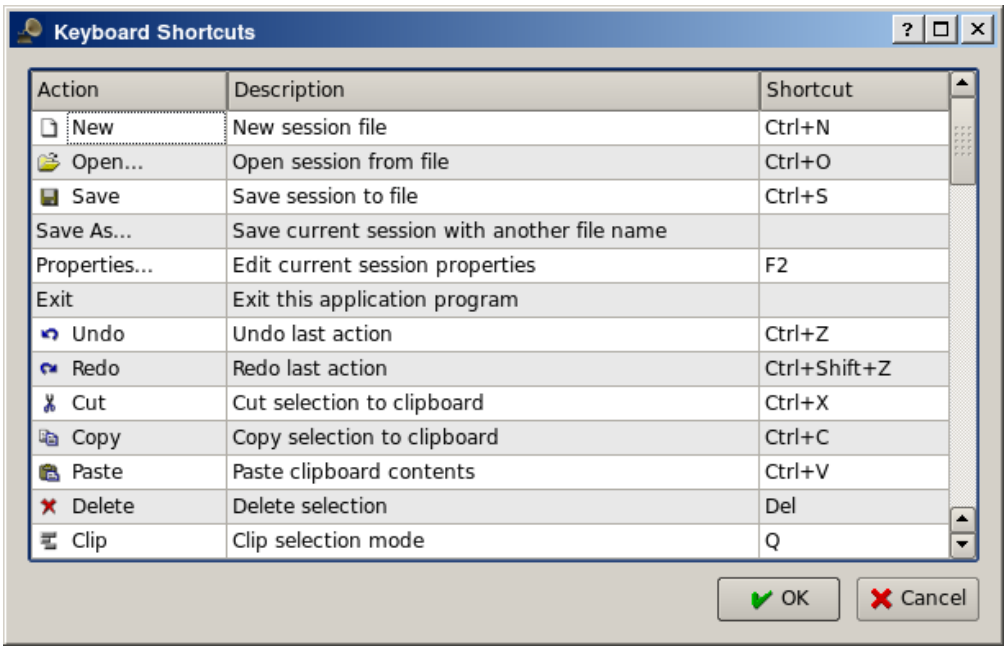


Figure 21: The Keyboard Shortcuts Editor

It is very straight forward in its use. Simply find the item you want to create a shortcut for, left click in the shortcut cell, and type on the key you wish to be assigned to that function. You will then see whatever key or key combination you chose appear in the context.

21. Menu Items

21.1 File Menu

New: Creates a new Session Project.
Open: Opens a previously created Session Project.
Open Recent: Contains a list of several of your last used Session Projects.
Save: Saves your current Session Project
Save As: Saves your current Session Project with naming conventions
Properties: Opens the Session Properties dialog
Exit: Exits the program.

21.2 Edit Menu

Undo: Will undo the last action.
Redo: Will Redo the last action.
Cut: Deletes and copies the item to the clipboard.
Copy: Copies the item to the clipboard
Paste: Pastes the item from the clipboard.
Delete: Deletes the selected item.
Select Mode: Selects the edit mode, one of Clip, Range or Rectangle.
Select: None, Range, Track or All
Edit Clip: Opens the Clip Editor dialog

21.3 Track Menu

Add Track: Will add either Audio or MIDI track.
Remove Track: Removes a track.
Track Properties: Calls the Track Properties dialog.
Import Tracks: Import either Audio or MIDI track.
Export Tracks: Export either Audio or MIDI track.

21.4 View Menu

Menu Bar: Will toggle whether the menu bar is shown.
Status Bar: Will toggle whether the Status Bar is shown.
Tool Bars: Toggles various Tool Bars on and off.
Files: Toggles whether the Files dialog is on or off.
Messages: toggles whether the Messages dialog is on or off.
Connections: Displays the Connections dialog.
Instruments: displays the Instruments dialog.
Buses: Opens the Buses Editor dialog.
Options: Opens the Program Options dialog.

21.5 Transport Menu

Backward:
Rewind:
Fast Forward:
Forward:
Loop:
Loop Set:

Play:
Record:
Metronome:
Follow Playhead:
Auto Backward:
Continue Past End:

22. Future Thoughts

As of its current status, there are many and rather fundamental functionality still missing that tear Qtractor apart from a finished product, let alone for the quest of its own goals. It's still a work in progress. In my own personal agenda priority, the following are the ones for taking care in times to come, in no special order:

General

- DSSI and Native VST Plugin Support
- LV2 Plugin Support
- Snap/Zoom Menu Accessibility
- Buffer/CPU% and Under-Runs Monitoring
- OSC or D-BUS Interface
- Integrated Scripting (angelscript?)
- File/Session Project Management
- File Error Message Handling
- Redesigned Session Project Dialog
- Session Project Templates

MIDI

- JACK-MIDI Support
- MIDI SysEx Librarian
- MIDI Event List and Filter
- MIDI Editor Draw Mode
- MIDI Groove/Swing Quantize
- MIDI Controller Feedback
- MIDI Sync (MTC/SMPTE, MIDI Clock)
- Automation / Dynamic envelope curves

Tracks

- Auto Cross-Fading of Overlapped Clips
- Punch in/out and Loop Recording (eg. Takes)
- File Revision Resource Management

Clips

- Clip Split Command
- Clip Locking, Muting
- Clip Plugins
- Pitch-Shifting of Individual Clips
- Paste Repeat Command
- Clip Linking

Mixer

- Mixer Presets
- Mixer Fader Groups
- Effect Aux Send/Returns (pseudo-plug-in)

23. Conclusion

As fundamental as is, Qtractor might be just some clone of earlier and existing software, being blatantly one of the Cakewalk's [11] Pro Audio series. It is however more than that, when regarded from the free open-source software development point of view, much like some cauldron framework, user-oriented, programmable, pattern sequencer, eventually targeted as a potential toolbox and workbench for easy, direct, live music-making and experimentalism

24. Acknowledgements

I am grateful to the free software open-source community in general and to the Linux Audio developers and users in particular, who dedicated their valuable time to the development and support of free audio and MIDI software, being Qtractor just one humble manifestation of such class of human endeavor.

Qtractor logo/icon is an original work of Andy Fitzsimon, borrowed from the public domain openclipart.org gallery.

25. References

[1] **Qtractor** - An Audio/MIDI multi-track sequencer
http://qtractor.sourceforge.net/
http://sourceforge.net/projects/qtractor/

[2] **Qt 4** - C++ class library and tools for cross-platform development and internationalization.
http://www.trolltech.org/products/qt/

[3] **JACK** Audio Connection Kit
http://jackaudio.org/

[4] **ALSA** - Advanced Linux Sound Architecture
http://www.alsa-project.org/

[5] **libsndfile** - C library for reading and writing files containing sampled sound
http://www.mega-nerd.com/libsndfile/

[6] **libvorbis** - Ogg Vorbis audio compression
http://xiph.org/vorbis/

[7] **libmad** - High-quality MPEG audio decoder
http://www.underbit.com/products/mad/

[8] **libsamplerate** – The secret rabbit code, C library for audio sample rate conversion
http://www.mega-nerd.com/SRC/

[9] **LADSPA** - Linux Audio Developer's Simple Plugin API
http://www.ladspa.org/

[10] **QjackCtl** – JACK Audio Connection Kit - Qt GUI Interface
http://qjackctl.sourceforge.net/
http://sourceforge.net/projects/qjackctl/

[11] **Cakewalk** - Powerful and easy to use products for music creation and recording
http://www.cakewalk.com/
ftp://ftp.cakewalk.com/pub/InstrumentDefinitions/

[12] **SoundTouch** – Sound Processing Library
http://www.surina.net/soundtouch/

[13] **rncbc.org** - My personal web presence, blog, forum and other mundane material
http://www.rncbc.org/

26. Tutorials