

Qtractor Manual & How-To's

[Qtractor Wiki](#)

mcbbc *et al.* 2025-10-27 11:00+0000

Home

Welcome to the Qtractor wiki!

This is an amalgamation of various existing documents, originally wiki-fied by [junkyardsparkle](#) and [Yassin Philip](#), with updates by several others. The current working pages are as below:

[Manual - Table of Contents](#)

[Manual - 1 Introduction](#)

[Manual - 2 Installing and Configuring Qtractor](#)

[Manual - 3 Learning Qtractor—An Example Session](#)

[Manual - 4 Qtractor—An Overview](#)

[Manual - 5 Qtractor Menus](#)

[Manual - 6 Appendixes](#)

The [How-To's](#) section, started by [Sean Beeson](#), contains the following pages:

[How To - 1 Compile Qtractor](#)

[How To - 2 Create Individual MIDI and Audio Buses-Ports](#)

[How To - 3 Set the number of channels an audio track records](#)

[How To - 4 Sample MIDI Composition Workflow](#)

[How To - 5 Sidechain Workflow with Carla Plugin](#)

[How To - 6 Alternative Sidechain Workflow with Aux Sends](#)

[How To - 7 Equal Latency for Tracks and Buses](#)

[How To - 8 Controlling a Plugin's Parameter from a MIDI Clip](#)

[How To - 9 Distributing Plugins' Load to multiple CPU Cores](#)

[How To - 10 Get SOLO functionality on Buses](#)

[How To - 11 Prevent Color Picker Freezing Qtractor](#)

[How To - 12 Automate Buses Experimentally with MIDI filters](#)

[How To - 13 Make Successful Audio Connections](#)

[How To - 14 Automate Buses and Tracks by Layer with Insert Controller](#)

[How To - 15 Use Multiple SoftSynths on a Track](#)

[How To - 16 Obtain a Multi Ghost Reference](#)

[How To - 17 Get Individual Drum Instruments with a MIDI Track](#)

[How To - 18 Take Advantage of Qtractor's Hidden Tricks](#)

Unofficial resources:

[Unofficial tutorials that are worthwhile](#)

Older manuals, which contain some different information to the wiki, can be found here:

- 0.3 and 0.5 user manuals, wiki-fied by [junkyardsparkle](#):

[qtractor 0.3.0 manual pt1](#)

[qtractor 0.3.0 manual pt2](#)

[qtractor 0.3.0 manual pt3](#)

[qtractor 0.5.x Part 0. Installing Qtractor](#)

[qtractor 0.5.x Part 1. Quick Start](#)

- The latest [Qtractor User Manual](#) from [Seth Kenlon \(not\)klaatu](#), as worked on by [Phil CM](#)

Manual Drafts of features for future versions can be found here:

- [Manual Draft - Loop Mode Recording](#) Recording multiple takes using loop mode recording (now added to [wiki](#))

[junkyardsparkle](#) also started the [Meta - Discussion] page, for general discussion of Qtractor documentation creation.

The wiki uses [Markdown](#) syntax.

Manual - Table of Contents

This wiki version of the manual is based on a document originally authored by Rui Nuno Capela, James Laco Hines and Stephen Doonan.

Table of Contents

[1. Introduction](#)

- [1.1. Abstract](#)
- [1.2. Introduction](#)

[2. Installing and Configuring Qtractor](#)

- [2.1. Compiling Qtractor](#)
- [2.2. Build dependencies](#)
 - [2.2.1. Mandatory packages](#)
 - [2.2.2. Optional packages](#)
- [2.3. Downloading Qtractor](#)
 - [2.3.1. Official releases](#)
 - [2.3.2. Qtractor for the experienced and adventurous](#)
- [2.4. Compiling and installing Qtractor](#)
 - [2.4.1. Standard compilation and installation](#)
 - [2.4.2. Compiling Qtractor with debugging support](#)
- [2.5. Qtractor's Configuration Settings File](#)
- [2.6. QJackCtl](#)
- [2.7. Audio and MIDI Input](#)

[3. Learning Qtractor—An Example Session](#)

- [3.1. Preparation](#)
- [3.2. Importing an Audio File](#)
- [3.3. Connecting the MIDI data source to Qtractor](#)
- [3.4. Creating a MIDI track](#)
- [3.5. Recording and Playback](#)
- [3.6. Next Steps](#)

4. Qtractor—An Overview

- [4.1. Routing: Connections, Ports, Tracks and Buses](#)
 - [4.1.1. Routing: General Concepts and Information](#)
 - [4.1.2. Routing in Qtractor](#)
 - [4.1.3. Routing: Technical Notes](#)
- [4.2. Qtractor's Main Window and Work Area](#)
- [4.3. Understanding a Qtractor Session \(recording or editing\)](#)
 - [4.3.1. Session Audio Sample Rate](#)
 - [4.3.2. Session Properties](#)
 - [4.3.3. Session Options](#)
- [4.4. Files](#)
- [4.5. Clips](#)
 - [4.5.1. Clips Summary](#)
 - [4.5.2. Copying, Cutting, Splitting and Merging](#)
 - [4.5.3. Fades and Cross-Fades](#)
 - [4.5.4. Clip Properties](#)
- [4.6. Qtractor Main Workspace: Tracks Area](#)
 - [4.6.1. Tracks Summary](#)
 - [4.6.2. Track States](#)
 - [4.6.3. Making Selections: Select Modes](#)
 - [4.6.4. Making Selections: Edit Markers](#)
 - [4.6.5. Loops](#)
 - [4.6.6. Paste Repeat](#)
 - [4.6.7. Punch In / Out](#)
 - [4.6.8. Automation](#)
- [4.7. Mixer](#)
 - [4.7.1. Mixer Summary](#)
 - [4.7.2. Setting Levels](#)
 - [4.7.3. Panning](#)
- [4.8. Signal Flow](#)
 - [4.8.1. Signal Flow Summary](#)
 - [4.8.2. Connections](#)
 - [4.8.3. Buses](#)
 - [4.8.4. Aux Sends](#)
 - [4.8.5. Inserts](#)
 - [4.8.6. Connection Persistence](#)
- [4.9. Plugins](#)
 - [4.9.1. Plugins Summary](#)
 - [4.9.2. LADSPA](#)
 - [4.9.3. DSSI](#)

- [4.9.4. VST \(Linux Native\)](#)
- [4.9.5. CLAP](#)
- [4.9.6. LV2](#)
- [4.10. Working with MIDI](#)
 - [4.10.1. MIDI Summary](#)
 - [4.10.2. Importing MIDI Data](#)
 - [4.10.3. Creating MIDI Data with a MIDI Controller](#)
 - [4.10.4. Creating MIDI Data with the MIDI Editor](#)
 - [4.10.5. Additional MIDI Editor Features](#)
 - [4.10.6. MIDI Instruments](#)
 - [4.10.7. Soft Synths](#)
 - [4.10.8. Soft Synths as Plugins](#)
 - [4.10.9. Stand-alone Soft Synths](#)
- [4.11. Working with Audio](#)
 - [4.11.1. Audio Summary](#)
 - [4.11.2. Importing Audio Files](#)
 - [4.11.3. Recording Audio](#)
 - [4.11.4. Recording from Line-In](#)
 - [4.11.5. Recording from USB](#)
 - [4.11.6. Recording Takes](#)
- [4.12. Audio / MIDI Export](#)
- [4.13. Snapshots, Templates and Archives](#)
 - [4.13.1. Snapshots](#)
 - [4.13.2. Templates](#)
 - [4.13.3. Archives](#)
- [4.14. Other Qtractor Features](#)
 - [4.14.1. Metronome](#)
 - [4.14.2. Keyboard Shortcuts Editor](#)

5. Qtractor Menus

- [5.1. Main Workspace](#)
- [5.2. MIDI Editor](#)
- [5.3. View / Options](#)

6. Appendixes

- [6.1 References](#)

Manual - 1 Introduction

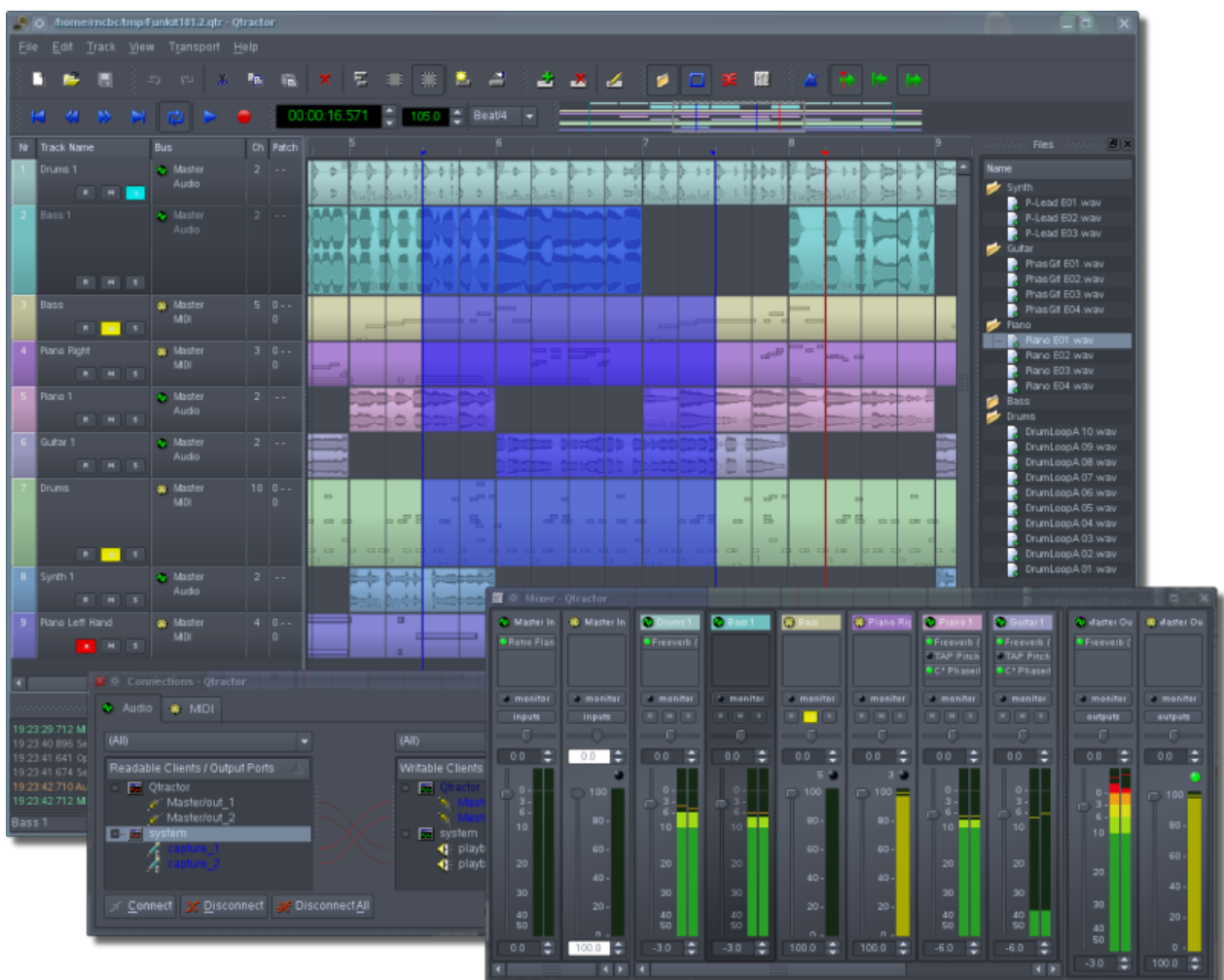
[Manual - Table of Contents](#)

1. Introduction

1.1. Abstract

Qtractor is a multi-track audio and MIDI recorder and editor. The program is written in C++ and uses the [Qt](#) toolkit for the GUI (graphical user interface) elements. Qtractor is free and open source software, licensed under the [GNU General Public License version 2](#) or later.

Qtractor runs exclusively under GNU/Linux and depends upon [ALSA](#) (Advanced Linux Sound Architecture) and [JACK](#) (the Jack Audio Connection Kit) to provide its MIDI and audio IO infrastructure. It currently has one developer, the originator of the project, Rui Nuno Capela. Development was started in April of 2005, initially as a Qt 3 application. Since October 2015, it is officially a Qt 5 application.



Main GUI window showing audio & MIDI tracks, Mixer & Connections windows

The project began as a sequencer intended mainly to be used with MIDI hardware devices but has since developed comprehensive support for plugins and gained moderately powerful audio

recording and editing facilities. It now aims to be a lightweight but reasonably powerful solution for hobbyist and semi-pro musicians. It does not aim to be a “do-it-all monolith [DAW](#)” (such as you may find in the commercial world), nor is it completely “modular” - rather, it should be considered something of a “hybrid”, which contains the kind of features that are most likely to be useful for its intended user-base. In the words of its developer, rather than a fully-fledged DAW, Qtractor is *a sequencer with DAW-like features*.

If your intention is to mimic the functionality of a recording studio, conducting a great deal of live audio recording and performing complex routing operations, something like [Ardour](#) may be more suited to your needs.

1.2. Introduction

Whilst Qtractor is still in its beta phase of development, it can already be comfortably used by hobbyists as a personal home recording studio or “bedroom studio”. It can record, import, arrange and edit both digital audio and MIDI data. Its interface will be familiar to users of other popular multi-track recording/editing applications.

In addition to recording digital audio and MIDI, Qtractor provides an environment for multi-track clip-oriented composing techniques common in modern music-making and aims to be intuitive and easy to use, yet powerful enough for the serious recording enthusiast.

Qtractor is a non-destructive audio editor. This means that your session’s audio files remain unchanged on disk after you have edited them. Instead, all edits to audio files are stored within the Qtractor session file.

As part of the GNU/Linux audio and MIDI ecosystem, Qtractor relies on other projects to provide functionality in various areas. Due to this approach, the user must consider his/her needs in areas such as synths, samplers, drum machines, effects processors and sounds (audio samples, sample packs (sf2/sfz/gig etc.), synth patches and so on) and use these other projects in tandem with Qtractor by referencing the information in this wiki. If you have no idea where to start and find the information in the wiki lacking, try searching online or checking sites such as the [Linux Audio Wiki](#) and [LinuxMusicians](#). There are also various projects on [Rui Nuno Capela’s site](#) which should enable you to get up and running reasonably quickly.

Manual - 2 Installing and Configuring Qtractor

[Manual - Table of Contents](#)

2. Installing and configuring Qtractor

2.1. Compiling Qtractor

If Qtractor is not available from your Linux distribution's package manager/repository and nor can you find a standalone package (such as a .deb package for Debian, Ubuntu or other Debian-based system or an .rpm package for Red Hat, Fedora, SUSE, etc.), then you will have to build it from source. Doing so requires you have both a C++ compiler (such as g++, the GNU C++ compiler) as well as several other dependencies, the most important of which are listed below.

Those with experience in compiling Linux/UNIX software using autoconf should find the build process straightforward. Building Qtractor is fairly easy but both the compilation process and setting up a build environment (a collection of programs necessary for compiling software) can be a bit daunting to those new to Linux. You don't need to be a programmer to compile software but you will at least need to know the basics of using the Linux command line and how to use your distribution's package manager.

2.2. Build dependencies

In order to compile Qtractor you are required to have at least all the packages listed under section 2.2.1 installed. Everything listed under section 2.2.2 is optional but recommended to get the most out of Qtractor.

Note that some Linux distributions (such as Arch) include the development files in their packages whereas other distributions split packages up into binary and dev components. For example, the ALSA library under Debian and Ubuntu is contained in the package libasound2 but its development files are contained within libasound2-dev.

If you are compiling Qtractor under a Debian or Ubuntu-based distro you can install all the required build dependencies with one command by running:

```
sudo apt-get build-dep qtractor
```

If it doesn't work try:

```
sudo apt install qt6-base-dev qt6-base-dev-tools qt6-tools-dev qt6-tools-dev-  
tools qt6-l10n-tools qt6-svg-dev libqt6svg6-dev libjack-dev libjack-jackd2-dev  
libasound2-dev libsndfile-dev libvorbis-dev libmad0-dev libz-dev  
libsamplerate-dev librubberband-dev libfftw3-dev libaubio-dev ladspa-sdk dssi-  
dev liblo-dev lv2-dev liblilv-dev libsratom-dev libsord-dev libserd-dev  
libgtk2.0-dev libgdkmm-2.4-dev
```

2.2.1. Mandatory packages

- **Qt** framework (core, gui, xml) - C++ class library and tools for cross-platform development and internationalization <https://qt.io/>
- **JACK** - JACK Audio Connection Kit <http://jackaudio.org/>
- **ALSA** - Advanced Linux Sound Architecture <https://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/>

2.2.2. Optional packages

- **libvorbis** (enc, file) - Ogg Vorbis audio compression <https://xiph.org/vorbis/>
- **libmad** - High-quality MPEG audio decoder <https://www.underbit.com/products/mad/>
- **libsamplerate** - Secret Rabbit Code; C library for audio sample rate conversion <http://www.mega-nerd.com/SRC/>
- **librubberband** - Rubber Band Audio Time Stretcher; audio time-stretching and pitch-shifting library <https://breakfastquay.com/rubberband/>
- **liblo** - Lightweight OSC implementation (required for DSSI GUI and NSM support) <http://liblo.sourceforge.net/>
- **DSSI** - API for soft synth plugins with custom user interfaces <http://dssi.sourceforge.net/>
- **VST-SDK** - Steinberg's Virtual Studio Technology plugin format <https://www.steinberg.net/>
- **LV2** - Audio plugin standard; extensible successor to LADSPA <https://lv2plug.in/>
- **liblilv** - Lightweight LV2 implementation stack <https://drobilla.net/software/lilv/>
- **libaubio** - Library for real time audio labeling <https://aubio.org/>

2.3. Downloading Qtractor

2.3.1. Official releases

Qtractor is considered to be in its beta stage of development but is already fully functional. The latest official releases are publicly available from the qtractor.sourceforge.net project web site:

<https://qtractor.org/>

2.3.2. Qtractor for the experienced and adventurous

The latest “bleeding-edge” source code can be obtained from the qtractor git repository with the following command:

```
git clone --recursive https://git.code.sf.net/p/qtractor/code qtractor-code
```

Then go to the just created `qtractor-code` source tree directory:

```
cd qtractor-code
```

You should now be ready to build and install Qtractor.

NOTE: After the initial checking-out of the source code, if you want to update to the latest version in future you can fetch/merge the latest changes by using the following command from within the `qtractor-code` directory:

```
git pull
```

2.4. Compiling and installing Qtractor

2.4.1. Standard compilation and installation

The installation procedure follows the cmake build system (make sure you have version 3.15 or higher). From within the Qtractor source directory, run:

```
cmake -B build .  
cmake --build build
```

and optionally, as root:

```
cmake --install build
```

2.4.2. Compiling Qtractor with debugging support

If you find and report a bug in Qtractor, you may be asked to provide some debugging information. This short guide will explain how to build Qtractor with debugging enabled, making it easier to pinpoint the problematic code.

Rebuild it all from scratch, with:

```
cmake -DCMAKE_BUILD_TYPE=Debug -B build  
cmake --build build
```

Enable core dumps in a shell session:

```
ulimit -c unlimited
```

From the same shell command line, run the program until it crashes. You'll see something like this in the output when it happens:

```
Segmentation fault (core dumped)
```

Locate the dumped core file. Depending on your environmental settings it might be named just `core`, or something like `core.1234` (1234 is the process-id number of the crashing program), located in the last directory where the program was.

Load the core dump file into gdb:

```
gdb ./build/src/qtractor /path/to/core
```

At the gdb prompt just enter:

```
gdb> bt
```

or:

```
gdb> thread apply all bt
```

2.5. Qtractor's Configuration Settings File

Qtractor keeps its settings and configuration file in:

```
$HOME/.config/rncbc.org/Qtractor.conf
```

Normally, there is no need to manually edit this file. Most of Qtractor's options can be adjusted from the Options window under the View menu.

2.6. QJackCtl

In order to use Qtractor, [JACK](#) must be running in the background. JACK is a kind of patchbay for Linux (technically a “sound server”), enabling you to route audio and MIDI data between Qtractor and other JACK applications.

Some distributions may configure Qtractor such that JACK starts automatically when Qtractor is launched; others may not. If you launch Qtractor from your application menu and receive JACK errors in the message log at the bottom of the main workspace, quit Qtractor, start JACK manually, then launch Qtractor again.

The easiest way to start JACK is with [QJackCtl](#) (“Q JACK Control”), a control panel and global, synchronized timecode display. If this is not installed, install it from your distribution's repository or from the project's site.

If necessary, launch QJackCtl and click the Start button. This should begin a global, synchronized timecode clock that will allow Qtractor to use all available inputs and outputs just like a studio mixing board.

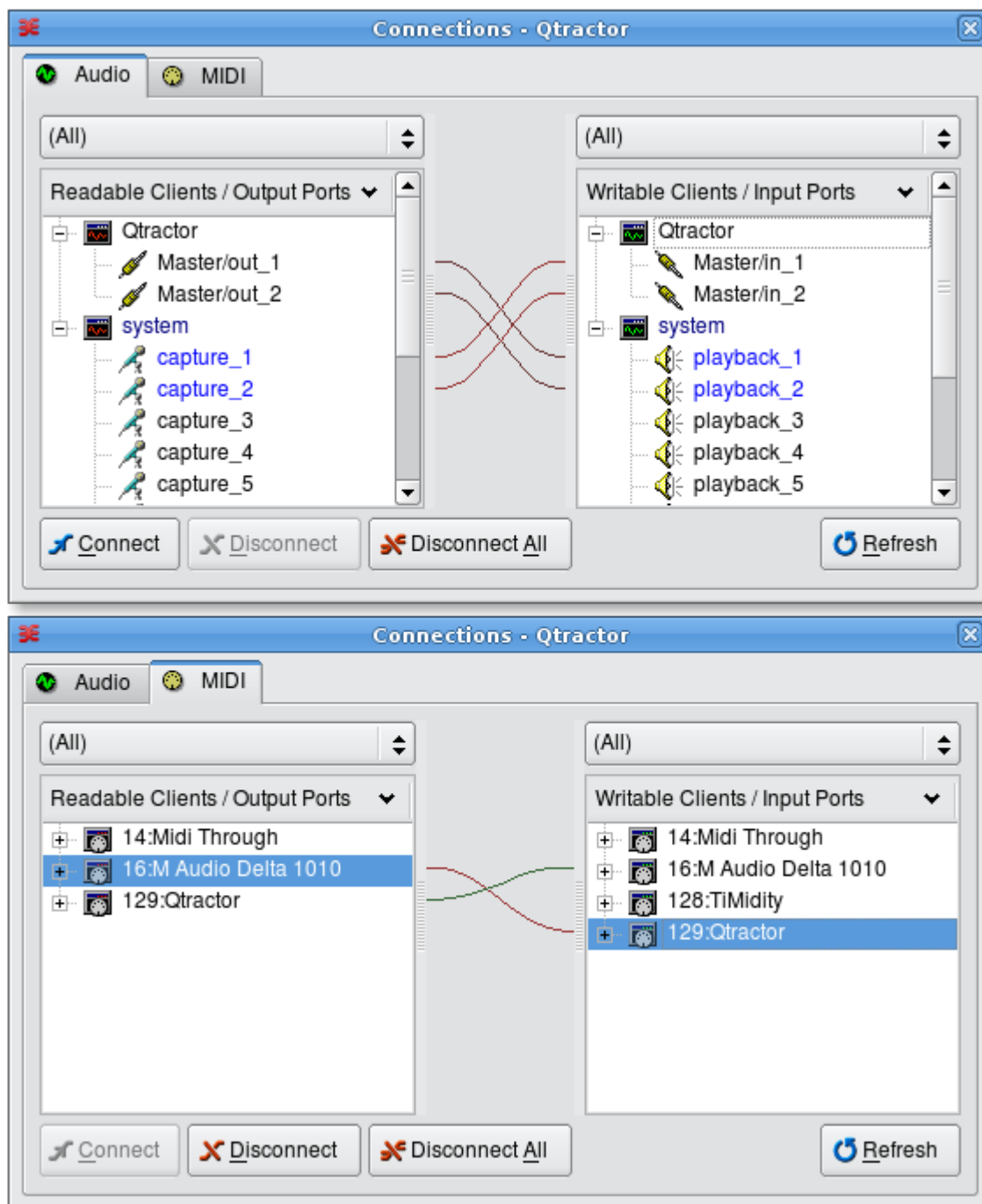


QJackCtl in operation

2.7. Audio and MIDI Input

Qtractor can record both digital audio and MIDI data, but it does not magically know what audio or MIDI data you wish to record; you must route audio and MIDI data to it manually. This can be done with QJackCtl, via the command line or using Qtractor's Connections window, pictured below.

Qtractor depends upon ALSA and JACK to utilize your audio hardware. It uses ALSA to communicate with MIDI hardware and JACK to route audio to/from various hardware and software “ports”.



Qtractor's Connections window, showing both the Audio and MIDI tabs. "Readable" ports are sources of data (where audio or MIDI data can come from, e.g. a microphone input or a MIDI keyboard controller) while "Writeable" ports are places that data can be routed to, such as a track, an audio output or a MIDI synthesizer

Manual - 3 Learning Qtractor–An Example Session

[Manual - Table of Contents](#)

3. Learning Qtractor–An Example Session

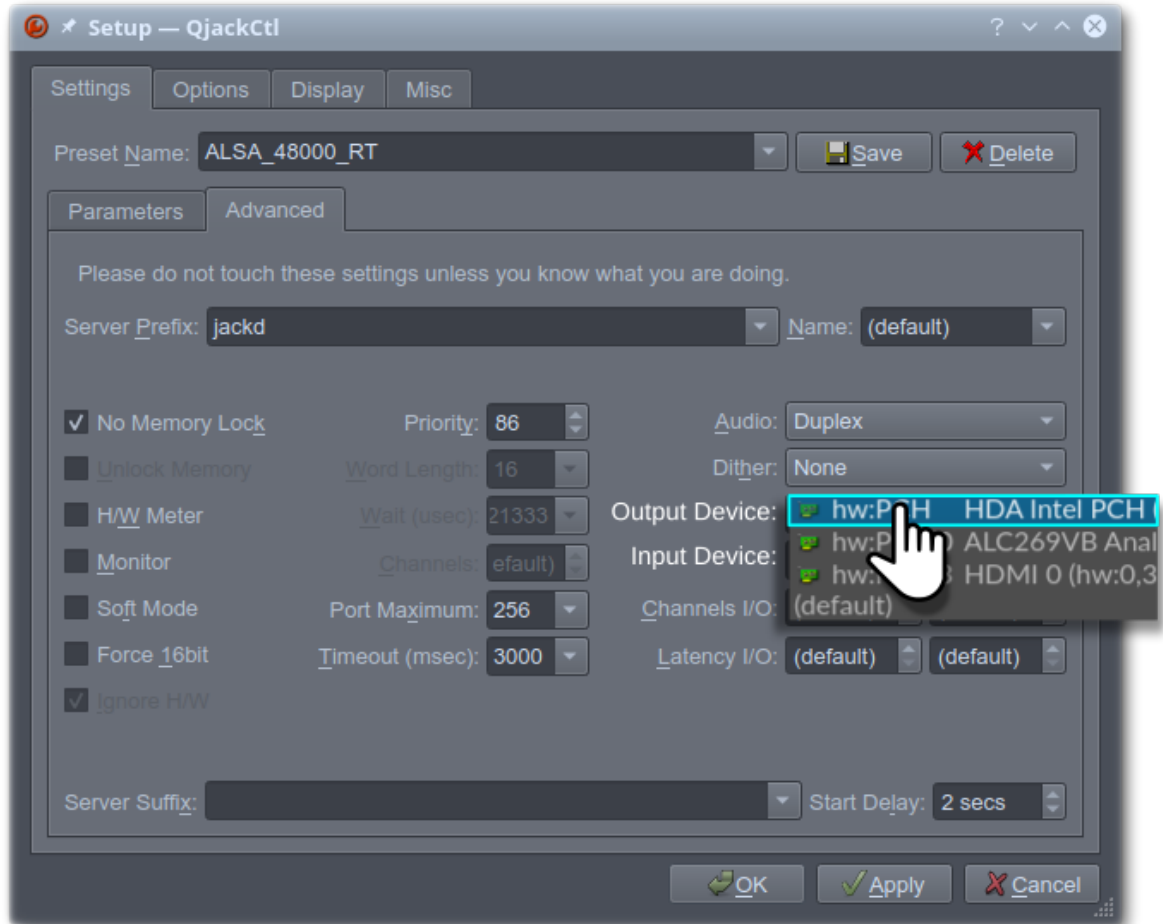
This chapter is written for those who may not be very familiar with digital audio recording or MIDI “sequencing” applications, or who wish to gain a quick overview of how Qtractor works. It can be used before exploring the program in greater depth and learning its features in detail. It describes an example Qtractor session and serves as a basic walk-through of the program.

3.1. Preparation

You’ll use Qtractor to import a pre-recorded audio file and record a MIDI track. A MIDI-triggered tone generator (in this case, a hardware tone generator outside the computer, although it could just as easily be a “soft synth” inside the computer) will produce the sound for the MIDI track as Qtractor plays it back.

To begin:

1. Start the [QJackCtl](#) application
2. Set the input and output audio hardware with QJackCtl if you need. This is the place and not Qtractor. You will find the options to do this at: **QJackCtl Setup > Configurations >**



3. Press the **Start** button to start JACK and launch Qtractor from your applications menu, dock, or launcher
4. It's good practise to **start every session by saving**. It might seem strange to save an as-yet empty session, but it's better to save an empty session than to start creating your masterpiece and have data files and MIDI files scattered all throughout your hard drive. Saving first is a good way to instantiate an environment in which you can keep all of your files and sounds organized and consolidated

To save, click the **File** menu and select **Save As**. This opens the [Session Properties](#) window.

In the **Name** field, name your session. For the **Directory** field, click the directory icon to create a new, empty directory for your session files and click **OK**. Save your session by clicking the **OK** button; a **Save Session** dialogue will open so that you can navigate to the directory you've created and name your session file, which will appear as a **.qtr** file. Click the **Save** button to confirm.

Now that the session has been saved, open the **Session Properties** window again using the menu item **File / Properties**. This time, click the **Properties** tab and set both the time signature and the tempo so that you can use the [metronome](#) to help you keep MIDI parts synchronized as you record them.

3.2. Importing an Audio File

The first thing you may like to do is import a pre-recorded audio file of drums and other percussion, to form the basis of the recording session. To bring audio into your project file, right-click in the *Files* pane on the right hand side of the main workspace and select *Add Files*, then choose the file or files you wish to import from the **Open Audio Files** window that appears. If the Files pane isn't visible, you can enable it by clicking on the yellow file icon, the one without an arrow above it. To place an audio file in a track, drag and drop it from the Files pane into the workspace. You can add it to an existing track, or drop it directly into an empty area and a new track will be created automatically.

3.3. Connecting the MIDI data source to Qtractor

Your MIDI piano-like keyboard normally routes its MIDI data (created when you strike the keys, for example) to its own internal tone generator, which then sounds like you're playing a real piano, or electric piano, harpsichord, bass guitar, etc. However, for this project in Qtractor you want to route the external keyboard's MIDI data to Qtractor. So, using a standard MIDI cable (or perhaps a MIDI to USB converter, depending on your set-up), connect the keyboard's output to the input of your sound card. Inside the computer, you will route the MIDI data from the sound card to a Qtractor MIDI input bus.

Next, open Qtractor's [Connections window](#) by clicking on the red icon near the top of the main workspace.



Connections icon

In the **Connections** window's **MIDI** tab, connect the MIDI output of the sound card (in the left pane, marked *Readable Clients / Output Ports*) to Qtractor's listing in the right pane (marked *Writable Clients / Input Ports*). If your sound card doesn't appear in the list, click the **Refresh** button in the bottom right to update the list of connections.

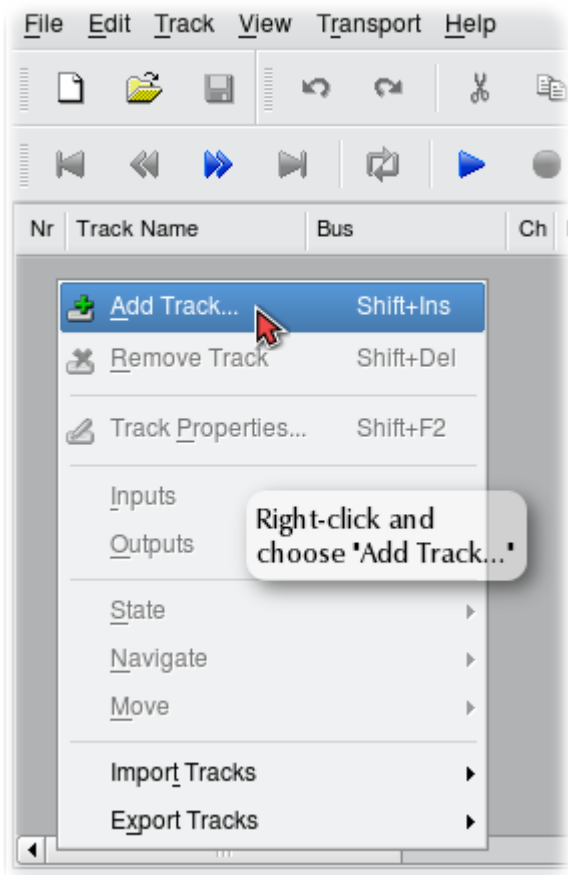
You can connect an output port (the source of the data, MIDI or audio) to an input port (the port that will receive the data) in several ways. One method is to left-click a port in the left pane, left-click a port in the right pane, then either left-click the **Connect** button or right-click elsewhere in the window and select *Connect* from the pop-up menu. Another method is to click a port in the left pane and, with the mouse button held down, drag the cursor to a port in the right pane until the port is highlighted, then release the mouse button. Regardless of the method you use, a line representing a "virtual cable" will appear between the two ports in the middle section of the window.

After connecting your sound card to Qtractor, you'll then need to connect Qtractor's output to your destination source, so that it can send the MIDI information back out again. Again in the **Connections** window, connect Qtractor's *Master/Out* listing in the left pane to your sound card in the right pane.

Once you've made your connections, close the Connections window by clicking its icon once more.

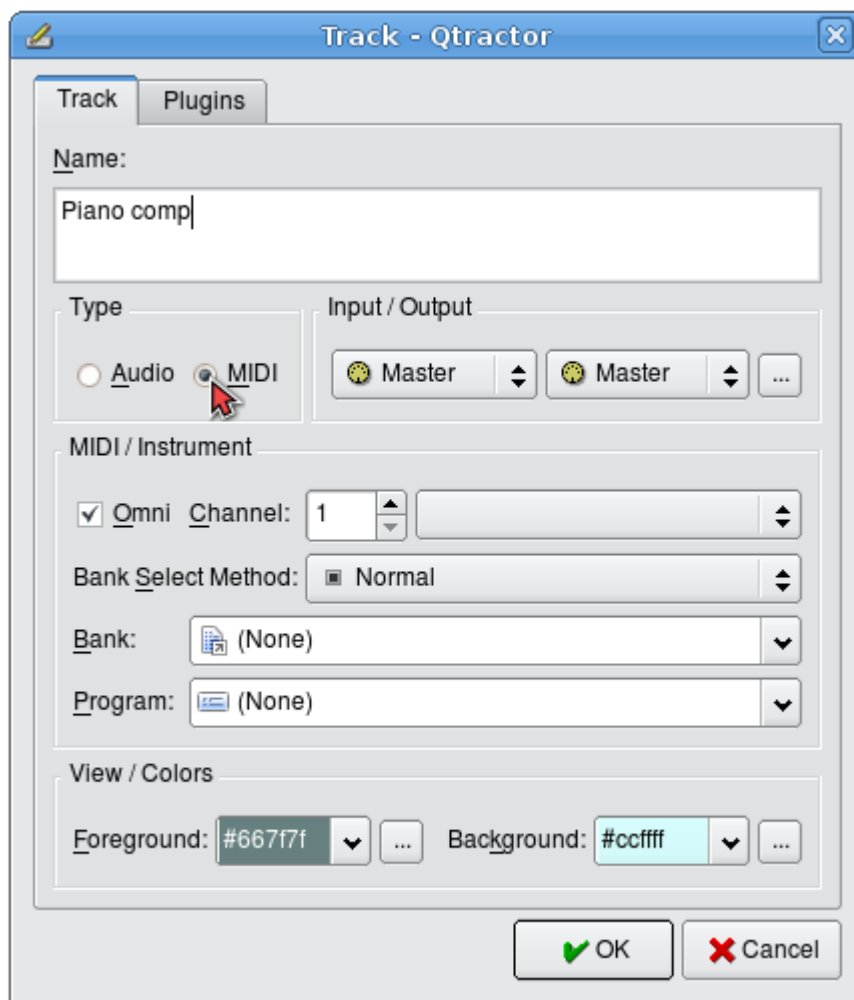
3.4. Creating a MIDI track

Now that Qtractor can receive and send MIDI data, it's time to create the first track in order to record that data. Right-click in the blank pane on the left of Qtractor's main window and, from the pop-up menu, choose *Add track...* You can also do this via the menu item **Track / Add Track...**



Adding a track

Qtractor's **Track Properties** window will then open.



Track Properties window

In the **Track Properties** window, click the **MIDI** radio button: you want this track to be used for MIDI, rather than audio, data. As soon as you click the MIDI button, the track is automatically connected to Qtractor’s Master MIDI input and output buses. These can be changed by using the drop-down lists and the ellipsis button in the **Input / Output** area of the Track Properties window, but there is no reason to do so right now. You can replace the default name “Track 1” with, for instance, “Piano comp.”

You can leave the MIDI **Channel** at the default “1” for now, as this channel designation is mainly for output. That is to say, when the track is played back and its MIDI data is sent to somewhere outside Qtractor (in this case, a hardware tone generator) that data will be sent on the MIDI channel specified here (there are 16 possible channels). However, you’re going to record now, so the eventual output MIDI channel doesn’t matter at the moment - it can easily be changed later before you play your MIDI tracks back.

What *does* matter however is the **Omni** checkbox. If enabled, Qtractor will record MIDI information received on *any* MIDI channel (omni means all). If disabled, it will record only information it receives on the channel number specified in the **Channel** setting, to the right of the Omni checkbox. Your MIDI keyboard might be set to transmit MIDI data on only one channel (e.g., MIDI channel 1), so you need to make sure that each of the MIDI tracks you plan to record on will receive data either from the correct channel, or from all channels. Click the **OK** button and a new empty track appears in Qtractor’s main window.

3.5. Recording and Playback

In order to record, you must arm the track by clicking on the “Record” button for that track. This is the “R” button that is present on every track strip, besides the other “M” and “S” buttons (respectively for “Mute” and “Solo” track state settings). You can also do this via the menu item **Track / State / Record**. Either way, the “R” button will turn red (or become highlighted, depending on your GUI settings) and the track is then set armed and ready for recording.

Next, turn on session recording mode by clicking on the Record button in the main transport toolbar (big red circle). You can also do this via the menu item **Transport / Record**.

Now you’re ready to record your first MIDI track. Take a little breath and... press “Play” (blue triangle next to the transport’s Record button), the space bar or choose the **Transport / Play** menu item: you’re rolling. Hit the MIDI keyboard and see a brand new MIDI clip taking shape while you’re performing. When done, press “Play” again to stop.

As previously mentioned, when playing back a MIDI track its data is sent on the channel number specified in the **Channel** setting of the **Track Properties** window. In order for your hardware tone generator to play back the MIDI data from Qtractor as actual sound, you’ll need to make sure that this channel tallies with your hardware set-up.

3.6. Next Steps

This chapter has been merely an introduction to what is a fairly far-reaching subject. More detailed information on the above topics, plus many other aspects of Qtractor, can be found in the [next chapter](#). Specifically, see [here](#) for more information on working with MIDI and [here](#) for audio. A sample MIDI workflow tutorial using a soft synth can be found in this [How To][How To - Sample MIDI Composition Workflow].

Manual - 4 Qtractor–An Overview

[Manual - Table of Contents](#)

4. Qtractor–An Overview

4.1. Routing: Connections, Ports, Tracks and Buses

4.1.1. Routing: General Concepts and Information

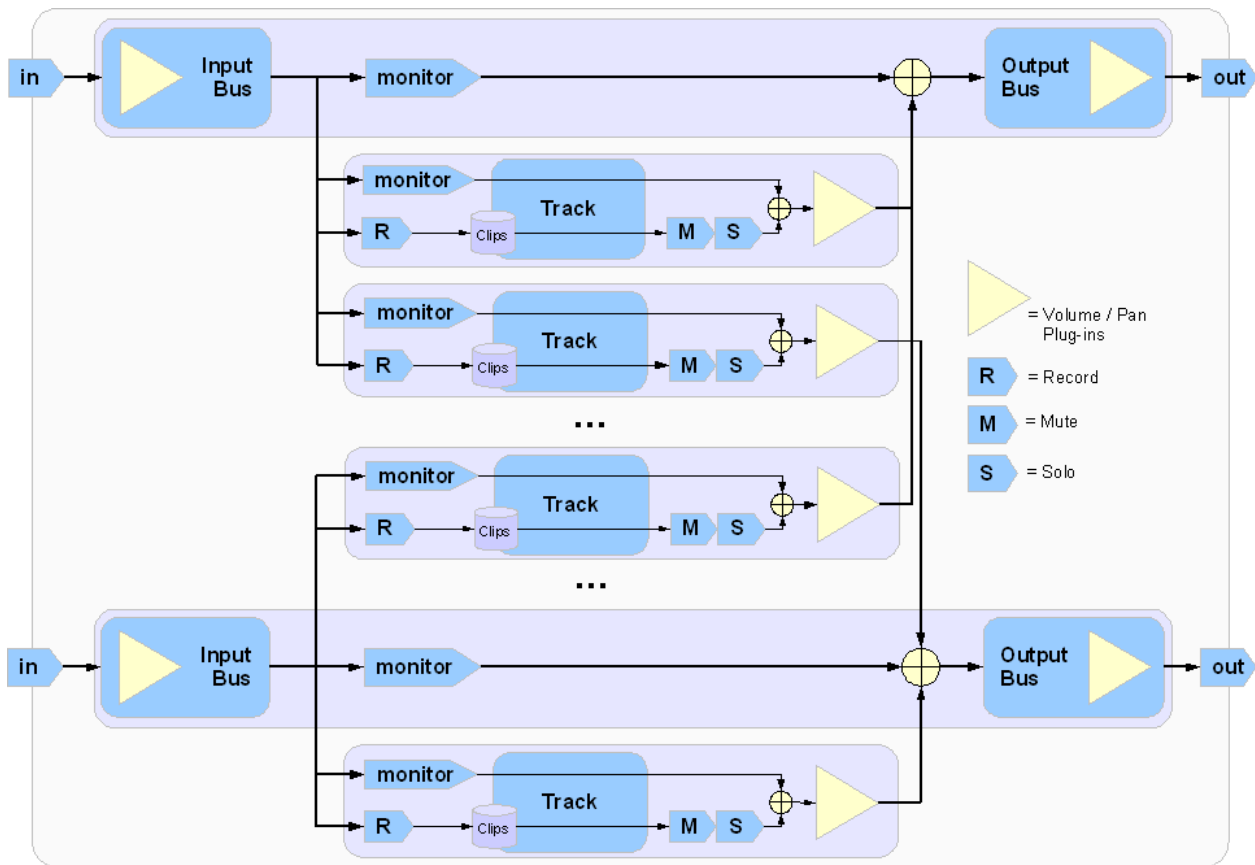
Qtractor can record and play digital data, specifically audio and MIDI data. To record, Qtractor must *get* the data from somewhere; to play the data, Qtractor must *send* it to somewhere that can understand the data stream and produce sound. The process of directing such data to and from – and through – software and hardware is called *routing*. In Qtractor, not much routing is done automatically. Most of it is left to the user, because it is the user's preferences, desires and goals in any specific project that determine how the audio and MIDI data should be routed.

In order to route data, one must have an understanding of the various connections and pathways that are available. A *bus* is like a pipeline through which one or more streams of data can travel. A *port* is like a valve at one or both ends of a bus. A port is normally closed, not allowing any data to flow through, but once a connection is made to it, it opens to allow data to pass through it from one bus to another. However, simply because a connection is made and the port is open doesn't mean that data is flowing through it: the data flow is initiated by the software or hardware at one or both of the ends of the bus.

A *track* can be thought of as a place where digital audio or MIDI data is deposited, rather than a pathway through which such data moves. Data can be deposited in various ways, such as recording, pasting or importing. Tracks are, however, still part of the signal or data flow, as they can “pluck” a copy of their data and send it to one or more buses (and, through those buses, to other destinations). In concrete terms:

- During recording, a track becomes the *destination* for audio or MIDI data entering Qtractor's input bus(es)
- During playback, a track becomes the *source* of audio or MIDI data, which is then sent through a Qtractor output bus to a destination, such as a sound card and its attached speakers

Below is a representation of the connections (ports), buses and routes in Qtractor through which audio or MIDI data can flow.



Audio & MIDI data flow, in, through and out of Qtractor

Routing in Qtractor is very flexible and therefore signal/data flow can become very complex. Data streams can merge and flow through the same bus, or split and go through separate buses to different destinations. They can likewise flow through different buses to the same destination, whilst passing through other intermediate software or hardware in which the data is possibly manipulated in some way. Because of this complexity it is beneficial to develop a solid understanding of the issues related to routing, the many possible routes that data can take and the many possible connections that can be made.

4.1.2. Routing in Qtractor

Routing can be accomplished in several ways, including:

- Using the [Connections window](#) to connect Qtractor's input and output buses to outside *sources* (for “reading” and, possibly, recording data) and *destinations* (for “writing” data to, for example, a sound card)
- Using the Track Properties window to assign a track's input and output [buses](#)
- Setting the [state](#) of a track, from among Record, Mute, Solo and Monitor (some of which are mutually exclusive)
- Adding [Plugins](#), [Aux Sends](#) or [Inserts](#) to tracks or buses. **Aux Sends** are used to route audio to *internal* destinations, i.e. Qtractor buses; **Inserts** are used for *external* destinations, i.e. applications outside of Qtractor

4.1.3. Routing: Technical Notes

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 to the central JACK real-time audio processing cycle, through a lock-free ring-buffer. These audio file ring-buffers are recycled (filled/emptied) at a one second threshold and have 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 an 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. Audio frame relocation is accounted from successive JACK client process cycles (i.e. buffer-period resolution).

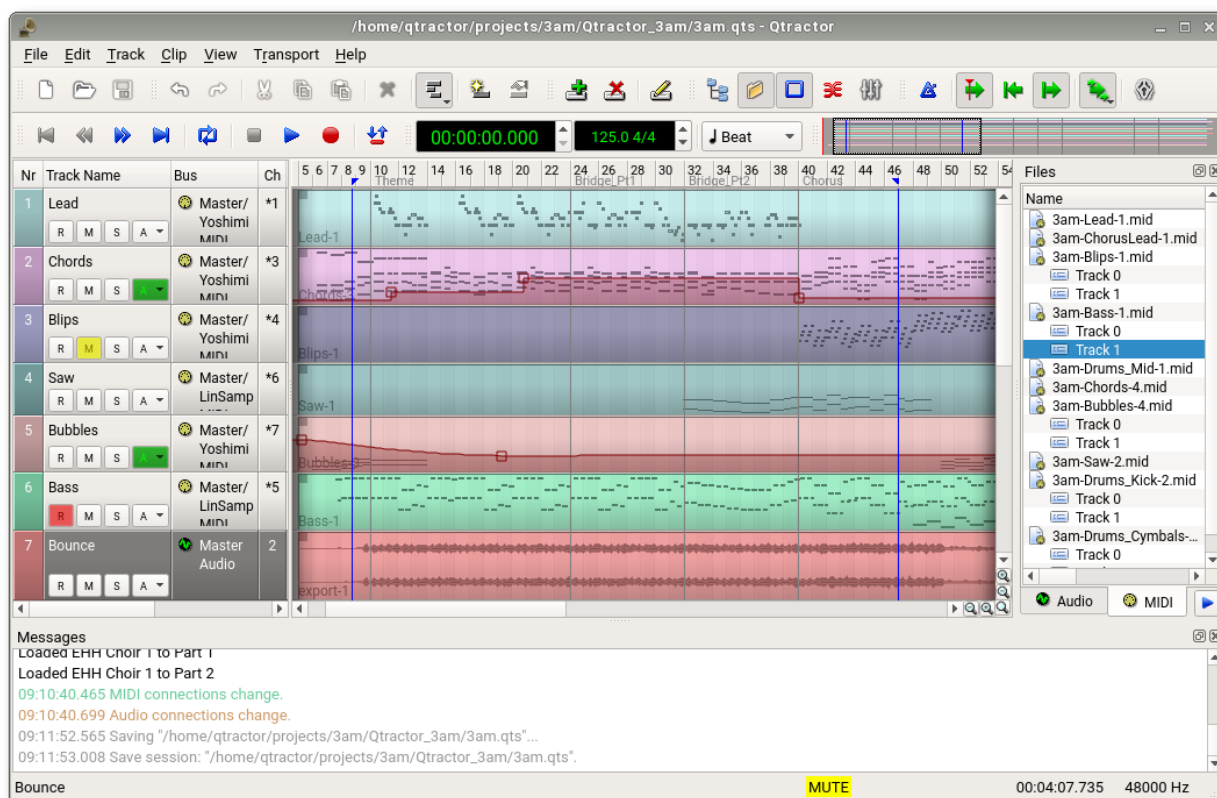
In this particular design, JACK and ALSA sequencer ports are logically aggregated as buses with respect to the audio and MIDI signal routing paths, functioning as fundamental device interfaces. Input buses, through exposing their respective input ports, are the inlets responsible for capture and recording. Output buses are the main signal outlets and are responsible for providing playback and, more importantly, mix-down connections.

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 bus determines the number of channels the track supports. Clips bound to disparate multi-channel audio files, for which the number of channels does not match the track's, are automatically resolved on mix-down. The illustration in the [above section](#) shows one typical signal flow block diagram.

4.2. Qtractor's Main Window and Work Area

Qtractor's graphical user interface follows a standard design of most modern digital audio and MIDI workstations. The interface is easy and intuitive enough to easily interact with in order to discover the potential of the underlying inner core of the application where its functionality is implemented.

The following illustration shows an overall view of the GUI with an example session loaded into the workspace.

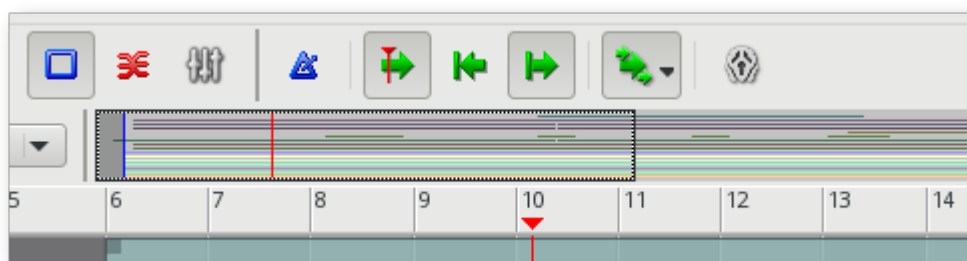


Qtractor's main window and work area

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

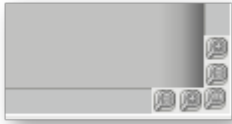
- **Top** - Menus and toolbar icons, with navigation window in bottom-right
- **Left** - Tracks area, showing name, state, buses etc.
- **Middle** - Audio and MIDI clips area, with timeline above
- **Right** - Files pane
- **Bottom** - Messages window

To move around the work area use the horizontal/vertical scroll bars at the bottom/right of the windows, or the mouse wheel to move up/down and **Shift**+mouse wheel to move left/right. You can also click on the navigation window (above the timeline) to jump to specific areas of the session.



The navigation window, situated between the main toolbar icons and timeline

To zoom in/out horizontally and vertically use the magnifying glass buttons in the bottom-right of the main view. **Shift**-clicking these allows you to zoom in greater increments and **Ctrl**-clicking will zoom to the maximum/minimum. You can also zoom in/out horizontally using **Ctrl**+mouse wheel.



“Magnifying glass” zoom buttons, situated in the bottom-right of the main view

The clips area in the middle is where most of the action takes place. It contains visual displays and representations of audio waveforms or MIDI data. This section is used for editing Clip objects (portions or all of any particular audio or MIDI file, recorded or imported) and for navigation within the project or “session.”

The very bottom-right of the screen shows the *Session total time* and *Session sample rate*. To the left of these is a panel which provides information on the session’s current state. From left to right, the indicators and their meanings are:

- **MOD** - session state has been modified since last save
- **REC** - session is set to record when transport next starts
- **MUTE** - at least one track is muted
- **SOLO** - at least one track is soloed
- **LOOP** - session is set to loop when transport next starts
- **XRUN** - an x-run has occurred



Session state, total time and sample rate information

Qtractor also has other useful windows, such as the Mixer window and the Connections (patch-bay) window. These can be opened via the **View / Windows** menu, or by pressing **F8** for Connections or **F9** for the Mixer. Three utility windows are additionally featured: the *Messages* window, for general information and debugging purposes; the *Files* pane, where audio and MIDI files are organised and selected; and the *File System* browser, which provides access to your file system in a manner similar to a file manager (such as Dolphin, GNOME Files, Thunar and so on).

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...*, available from the **View** menu in the main menu bar.

4.3. Understanding a Qtractor Session (recording or editing)

A Qtractor session project contains all the information about your Clips and their placement, Mixer set-up, plugins, tempo, time signature and Connections patch-bay. When creating or saving a project, all this and any related settings are saved on your hard disk within this session project file.

To create a new session, simply start Qtractor and begin working. If you already have a session loaded and want to start a new one, choose **File / New** or click the *New session* icon (blank piece of paper) in the upper-left of the screen. To open a session, either use the **File / Open...** menu

command or drag your session file (**.qtr** or **.qts**, which are essentially the same) into the left or middle pane.

4.3.1. Session Audio Sample Rate

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](#) server running at the time the session is created.

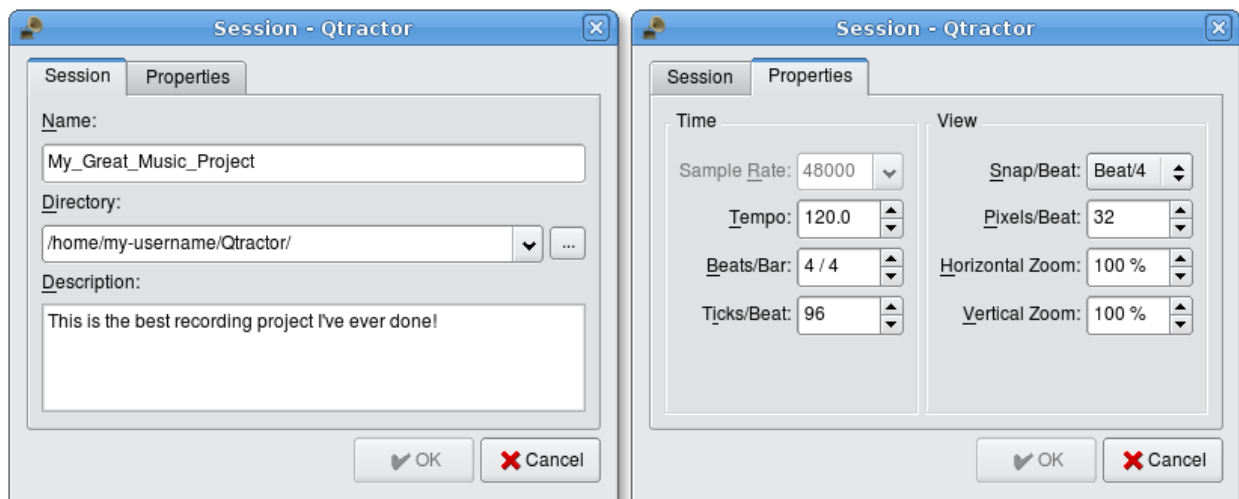
Any attempt to convert non-matching sample-rate sessions will result in a recommendation warning message.

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

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.

4.3.2. Session Properties

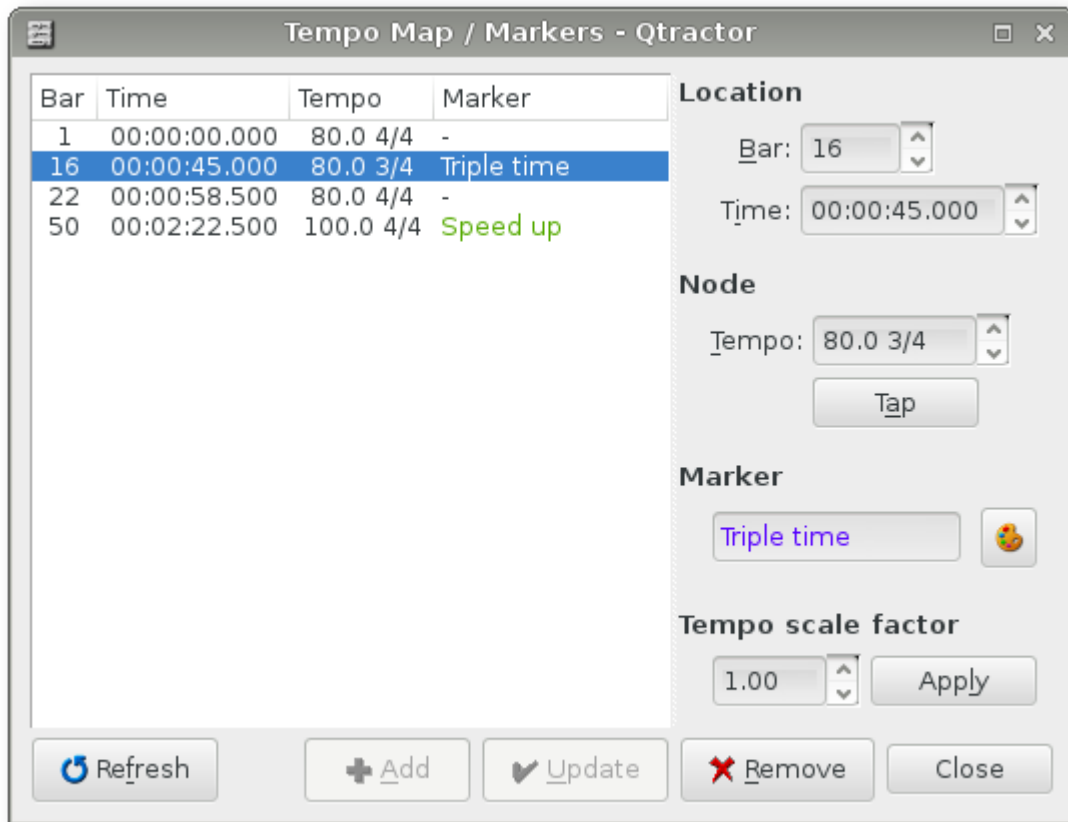
To access the session properties, choose the **File / Properties** menu item. Here, you can name your Qtractor session, set the tempo, time signature (how many beats per bar) and decide how many “ticks” (the smallest time unit in a session) will be within the time span of each beat. If you choose to rename your session, Qtractor will update any midi and audio clip file names to match this.



Session properties window showing both Session and Properties tabs

Tempo and time signature can be changed during the course of your composition using the **Tempo Map / Markers** window. This is accessed via the **View** menu or by double-clicking the region directly above the tracks which displays the bar numbers. Determine where you wish your tempo/time signature change to occur in **Location**, input the details of the change in **Node**, then click either **Add** to add a new node, or **Update** to make changes to a previously created one. If desired, you can also add descriptive text in **Marker**, plus a colour, by clicking the palette icon next to the text field. Markers may be assigned to tempo/time signature changes, or independent

of them. Once you're happy with the content of your Marker, click **Add** to add it to the map. The other option in this window is **Tempo scale factor**, which allows you to scale the entire Tempo Map (speeding up/slowing down the whole composition) by an arbitrary factor.

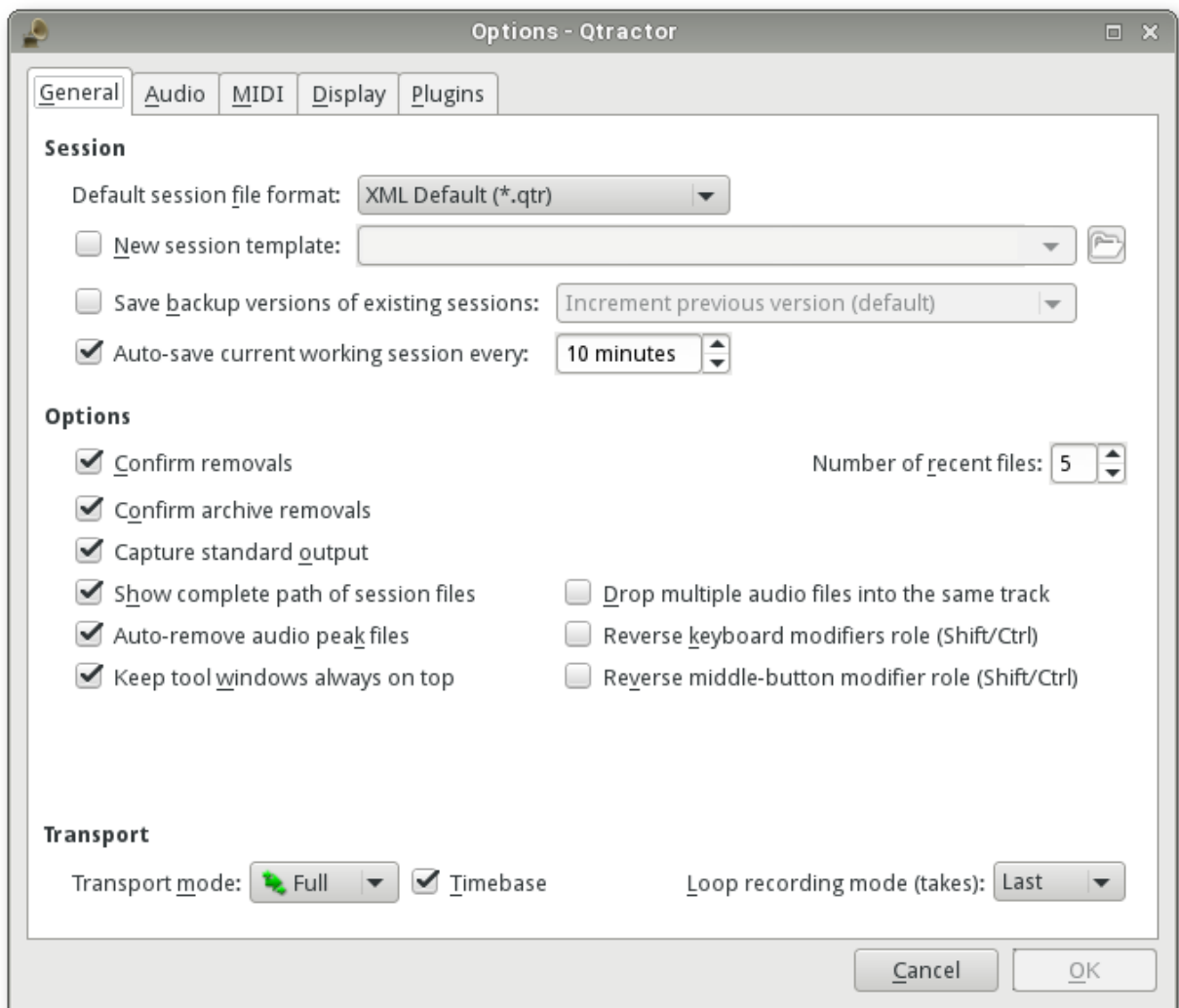


Tempo Map / Markers window showing tempo/time signature changes and markers

There are two limitations to be aware of with regard to tempo and time signature changes. Firstly, changes must occur at the start of a bar - there is no facility to add changes within one. If you, for example, need to insert a change into the middle of a 4/4 bar, create two 2/4 bars and add the change to the start of the second of these. Secondly, there is no facility to “ramp” between two different tempos (gradually increase/decrease tempo over a defined period). Hopefully, such a feature will be added in the future, though whether this will happen or not is currently unclear.

4.3.3. Session Options

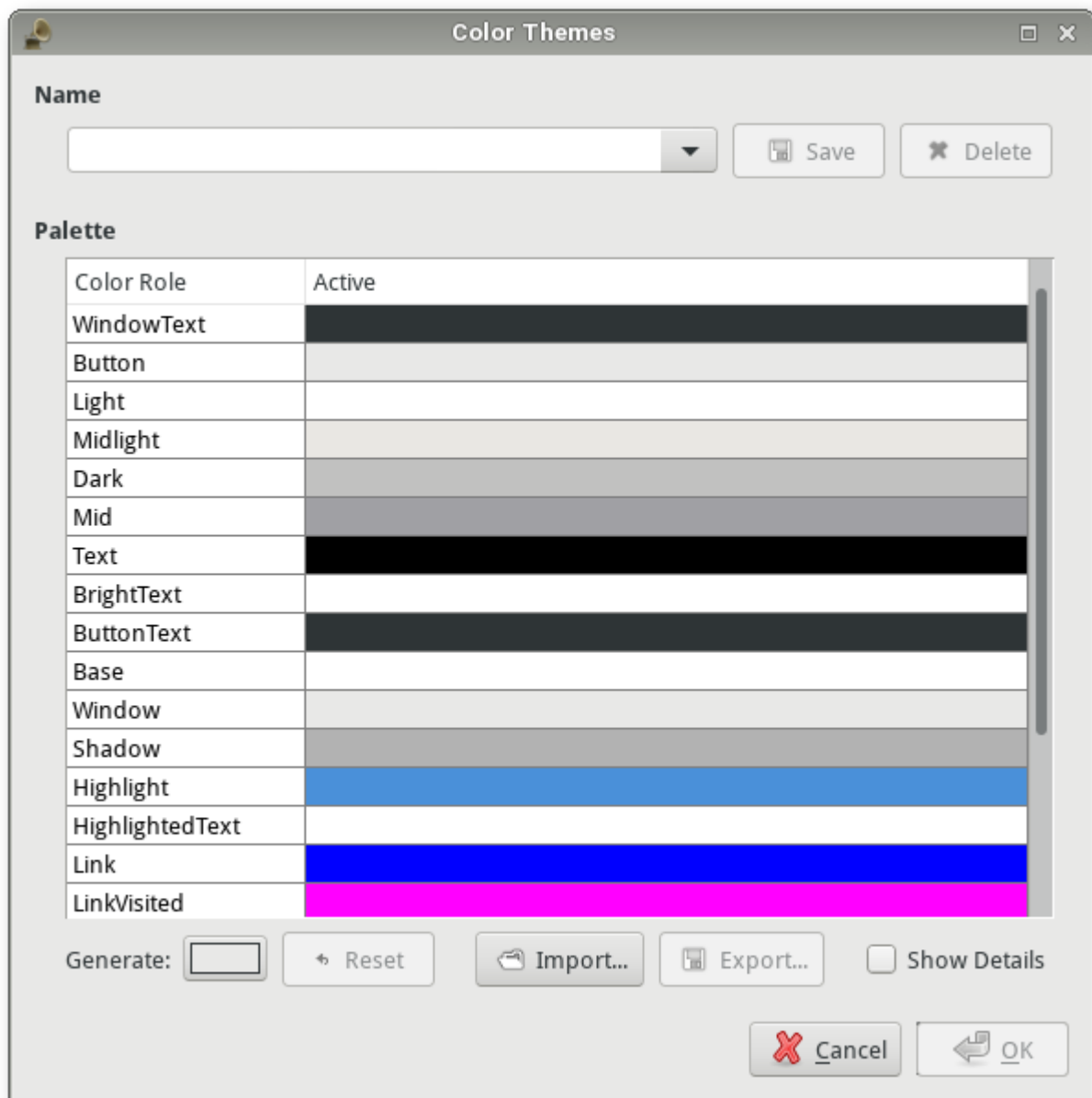
Access the Options windows via the **View / Options...** menu item. This window allows you to control the global parameters of Qtractor. These are global settings which are not saved within your session file.



Options window, showing General tab

The window is divided into five tabs: **General**, **Audio**, **MIDI**, **Display** and **Plugins**. Most of the settings are self-explanatory, and there are tool tips to guide you, but some which require more explanation are covered elsewhere in this chapter. The default options should largely be fine to begin with, but if you're having trouble seeing parts of the GUI you may wish to adjust the options in the **Display** tab. Qtractor takes its look-and-feel from your system settings by default, but on this tab you can disable *Use desktop environment native dialogs* (which should be off by default) and choose your preferred *Color theme* and *Style theme*. You may also wish to run `qtconfig` to perform configuration for all Qt applications (of which Qtractor is one).

If you need to make detailed colour-related changes, click the ... button to the right of *Color theme* to open the **Color Themes** window. Here, you can fine-tune a variety of settings, plus import them from/export them to external files via the **Import** and **Export** buttons at the bottom of the window.



Color Themes window, with the Show Details option enabled

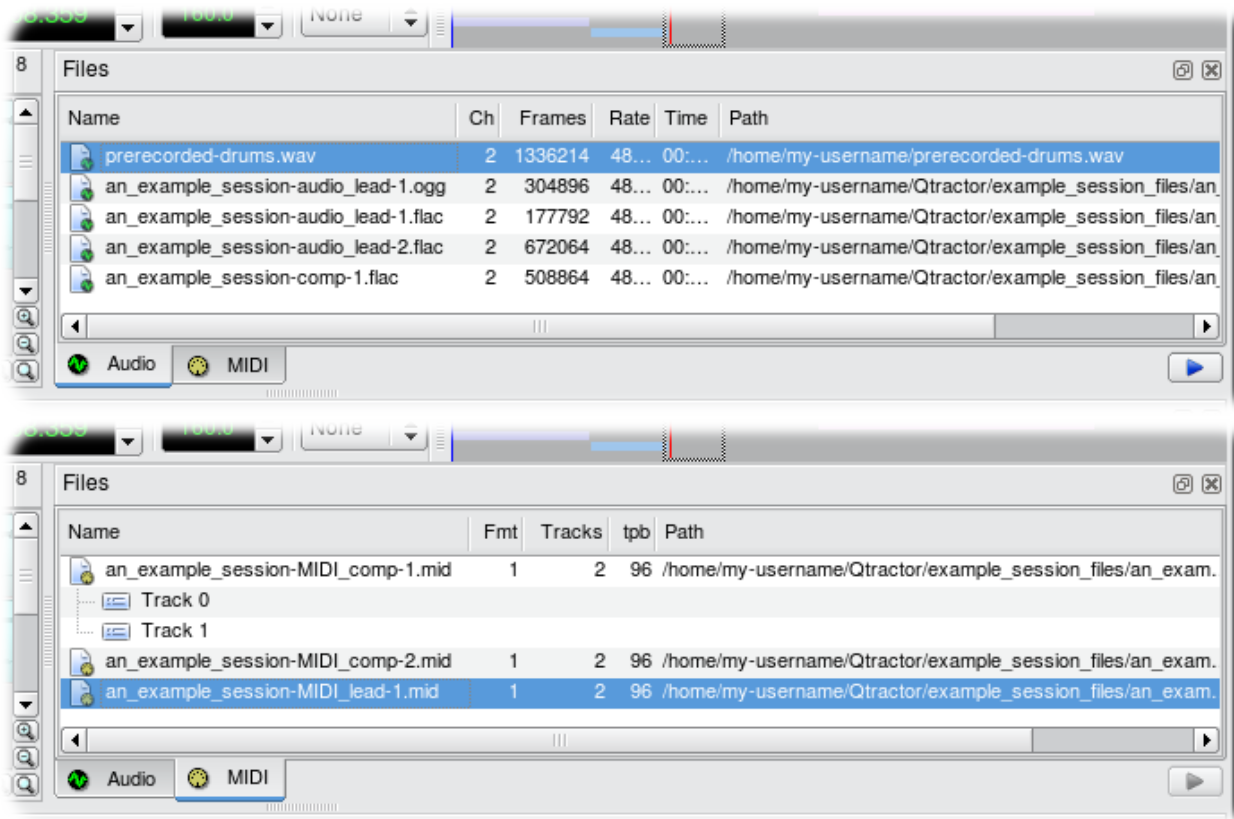
Finally, if you experience problems with Qtractor when using a desktop environment based on GTK+ (GNOME, Xfce etc.), try a Qt-based theme for *Style theme* and make sure that *Use desktop environment native dialogs* is disabled.

4.4. Files

Sound file selection is made available through a tabbed mini-organizer, known as the *Files* pane, which lists all the files contained in the current session. Audio and MIDI file lists are kept separate on their respective tabs. Audio file format support is provided by [libsndfile](#), enabling the use of wav, aiff, flac, au, etc. The optional libraries [libvorbis](#) and [libmad](#) provide support for ogg and mp3 formats respectively. MIDI file support covers the usual SMF formats 0 and 1, through a native, home-brew implementation.

Files can be imported using the menu command **Track / Import Tracks / Audio...** or **MIDI...**, or by right-clicking in the relevant tab of the Files pane and choosing **Add Files...**. If the Files pane is not visible, you can enable it via **View / Windows / Files**. The default position for the

Files pane is to the right of the clips area, but it can be detached and docked elsewhere (such as to the left of the Tracks pane) if needed. You can preview files by double-clicking them, or by selecting them and either clicking the play button on the lower right hand side of the pane or choosing **Play** from the right-click menu.



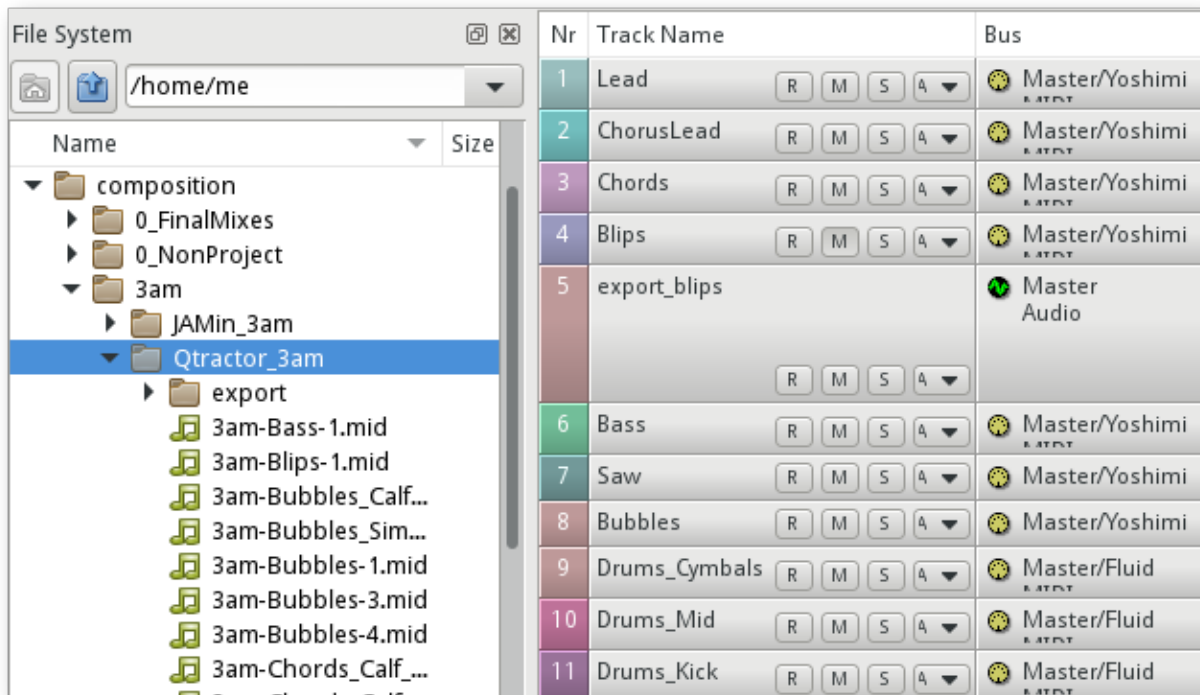
The Files pane of Qtractor's main workspace. Both Audio and MIDI tabs are shown

When importing a MIDI file, note that the file's tempo map will be reproduced in Qtractor *only* if it is the first MIDI content to be loaded into the current session. If the current session already has one or more MIDI tracks (even if the tracks contain no actual clips), the tempo map from the imported MIDI file will be discarded. This also means that if a **New session template** containing tracks has been enabled via the **View / Options...** / **General** menu, the tempo map of *any* imported MIDI file will be automatically discarded.

Individual and multiple files can be drag-and-dropped from the desktop environment and within the file list. Dropping a file into a blank region of the tracks area will create a new track and add the clip (which represents the file) to it. Dropping a file into an existing track will create a new clip within it. When dropping an audio file into an existing track you can drag the file itself, but with a MIDI clip you must drag your desired Track/Channel (displayed when clicking the file name in the Files pane), rather than the file itself. Finally, dragging a session file (**.qtr** or **.qts**, which are essentially the same) into the left or middle pane will load that session.

After working on a project for a while you'll probably find that the Files pane contains files which you no longer need. To remove anything not currently referenced in the tracks area, right-click the Files pane and choose **Cleanup**. Note that this merely cleans up the *list* of files - it does not delete the files themselves from your hard drive. If you wish also to remove non-needed files from your session directory, save the session as a **.qtz** archive and use this in place of your previous session. See [here](#) for details on archives.

For convenience, a File System browser with tree-view is also available. This can be enabled via **View / Windows / File System**. Its default position is to the left of the Tracks pane, but like the Files pane it can be detached and docked elsewhere. For instance, you may choose to have both windows on one side of the screen, with one above the other.



File System browser, docked to the left of the Tracks pane

4.5. Clips

4.5.1. Clips Summary

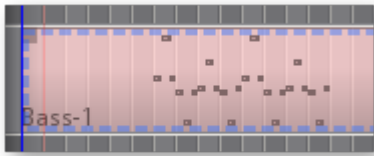
Clip objects are the elemental items of a session arrangement, and can contain either audio or MIDI data. A clip represents either a region of, or the entirety of, an actual audio or MIDI file. MIDI clip objects are representations of a sequence of events of one single MIDI channel, as extracted from an SMF format 0 file, or of one single track from an SMF format 1 file. Clips can be copied, truncated, lengthened/shortened (by dragging their edges) and time-stretched as if they were actual audio/MIDI data; editing a clip object in these ways is a non-destructive process.

Clips are placed on tracks either by importing audio or MIDI files as new tracks or by dragging and dropping files into the main track window. Empty clips may also be created by right-clicking on a track and choosing **Edit / Clip / New...**

4.5.2. Copying, Cutting, Splitting and Merging

To *copy* or *cut* a clip, select the clip, then right-click and choose **Copy** or **Cut** from the menu. Next, choose **Paste** from the right-click menu, position the box outline where you want to paste the clip and left-click. Note that when copying a MIDI clip within the same track, the two clips become *linked*, meaning that any change to one of them will automatically affect the other. If you select a clip which is currently linked to another, any such linked clips will be highlighted with a dashed outline, enabling you to see which other clips will be affected by changes made to the

selected one. To unlink a clip, thus disabling automatic changes, select it and choose **Clip / Unlink** from the right-click menu.



A linked clip highlighted with a dashed outline

To *split* a clip, select the clip, then position the playhead (red vertical line) where you want to split it and choose **Clip / Edit / Split** from the right-click menu. If you use this operation often, it is far more convenient to assign a keyboard shortcut for it. Since Qtractor never affects the original file it is referencing, any split you make to a clip is reversible. You can undo a split via the **Edit / Undo** menu, or by clicking and dragging the edge of the clip then extending it to its original length. Note that when dragging audio files to hide/reveal portions of them, the waveform may appear to “move” slightly. This is just a visual artefact - the position of the actual audio in the track is not affected.

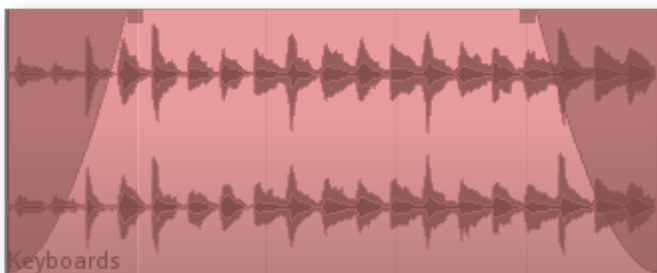
To *merge* clips and so consolidate a number of them into one file on your hard drive:

1. Select the clips within a track that you want to merge into one
2. Choose **Clip / Merge**, either via the main toolbar or the right-click menu
3. In the **Merge/Export** window, enter a new name for the merged file and save it to a logical location on your hard drive (preferably within your session folder)
4. The merged clip immediately replaces the old clips in your project

4.5.3. Fades and Cross-Fades

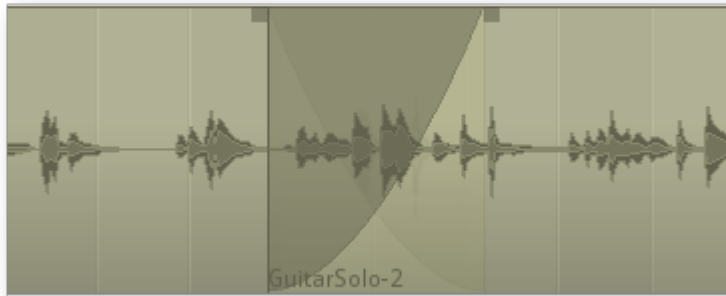
Fading a clip in or out is a common enough operation that there is a quick shortcut to achieve the effect. Notice that in the top corner of any audio or MIDI clip there is a semi-transparent square node. Clicking and dragging this node further into the clip will create a fade-in (if done at the beginning of the clip) or a fade-out (if done at the end).

The actual method of audio volume (gain) and MIDI velocity change can be one of linear, quadratic or cubic types. You can set your desired type via the clip’s **Properties** window (see next section).



An audio clip, with left and right upper corner fade handles drawn inward to create a fade-in and fade-out

When two clips overlap you may wish to cross-fade them, simultaneously fading-out the first and fading-in the second. You can do this manually, by dragging the fade handles to the desired lengths (they can still be manipulated even if clips overlap). Alternatively, you can select either clip and choose **Clip / Cross Fade** - Qtractor will then add a cross-fade, determined by the length of the overlap. Once you have your cross-fade you can fine-tune its length and type if needed, as per the information above. If you have trouble clicking on a particular clip because it's obscured by another, try selecting it from the [Files](#) pane.



Two audio clips cross-faded together

4.5.4. Clip Properties

A clip's properties can be accessed by double-clicking the clip (for audio clips), or by opening the clip and choosing **File / Properties** (for MIDI).

Audio clip properties include its Name (the label the clip is known by within the session); File path; Start time (location), Offset and Length; Gain; Panning; Format (Frames, Time or BBT); Fade-in/out length and type; Time Stretch percentage (can be set within a range of 10%-1000%) and options; and Pitch Shift amount (can be set within a range of -/+40 semitones). Time-stretching is applied in real-time at the buffering level, as a custom WSOLA algorithm based on and derived from the [SoundTouch](#) library.

MIDI clip properties include its Name; File path; Track/Channel; Start time, Offset and Length; Volume; Format (Frames, Time or BBT); and Fade-in/out length and type.

Clip - Qtractor

Name: AcousticGuitar-3

File: sition/guitarSession/guitarSession-AcousticGuitar-3.flac

Parameters

Start: 00:01:34.662 Gain: 0.0 dB

Offset: 00:01:53.601 Panning: 0.0

Length: 00:00:07.801 Format: Time

Fade In/Out

Fade In: 00:00:00.000 Quadratic 1

Fade Out: 00:00:00.226 Quadratic 2

Audio

Time Stretch: 100.0 % Pitch Shift: 0.00 semitones

☒ WSOLA time-stretching

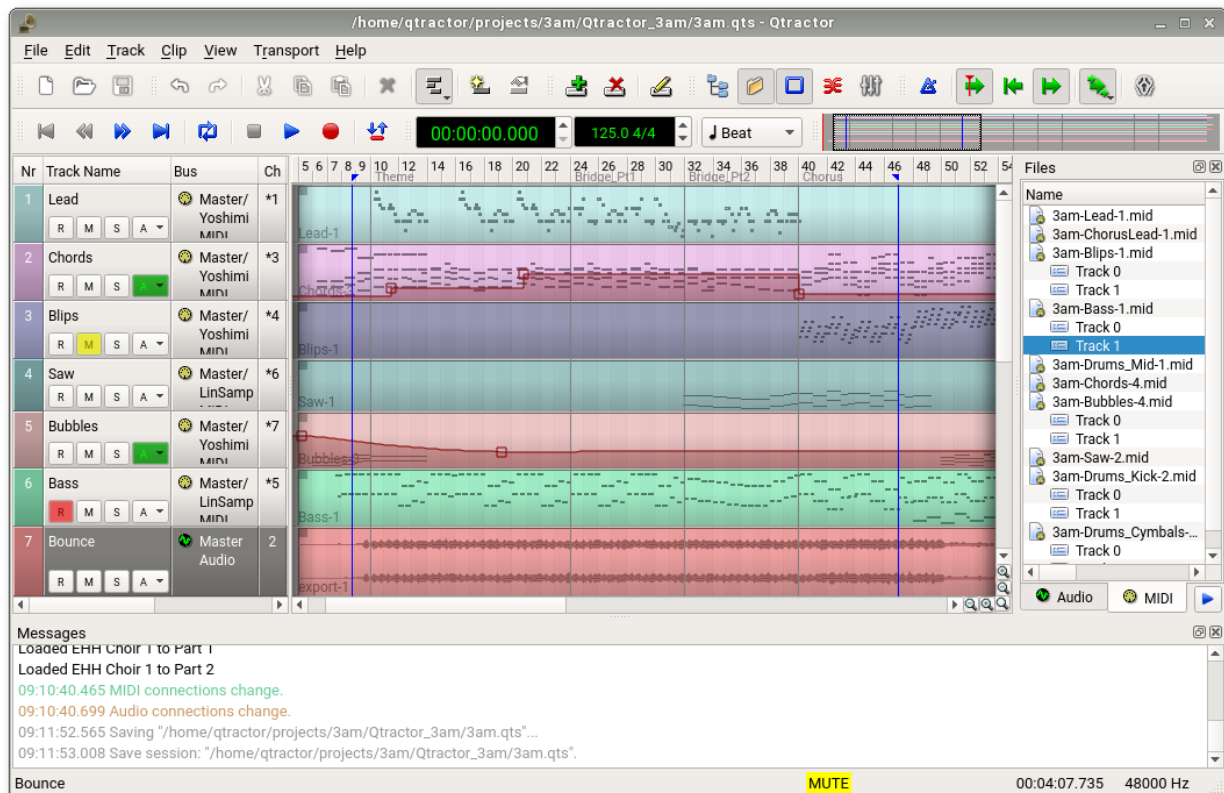
☐ WSOLA quick seek

Cancel OK

Audio clip properties window

4.6. Qtractor Main Workspace: Tracks Area

4.6.1. Tracks Summary



MIDI and audio tracks in the main workspace of Qtractor

Tracks are arranged as a sequence of one or more 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 performed here.

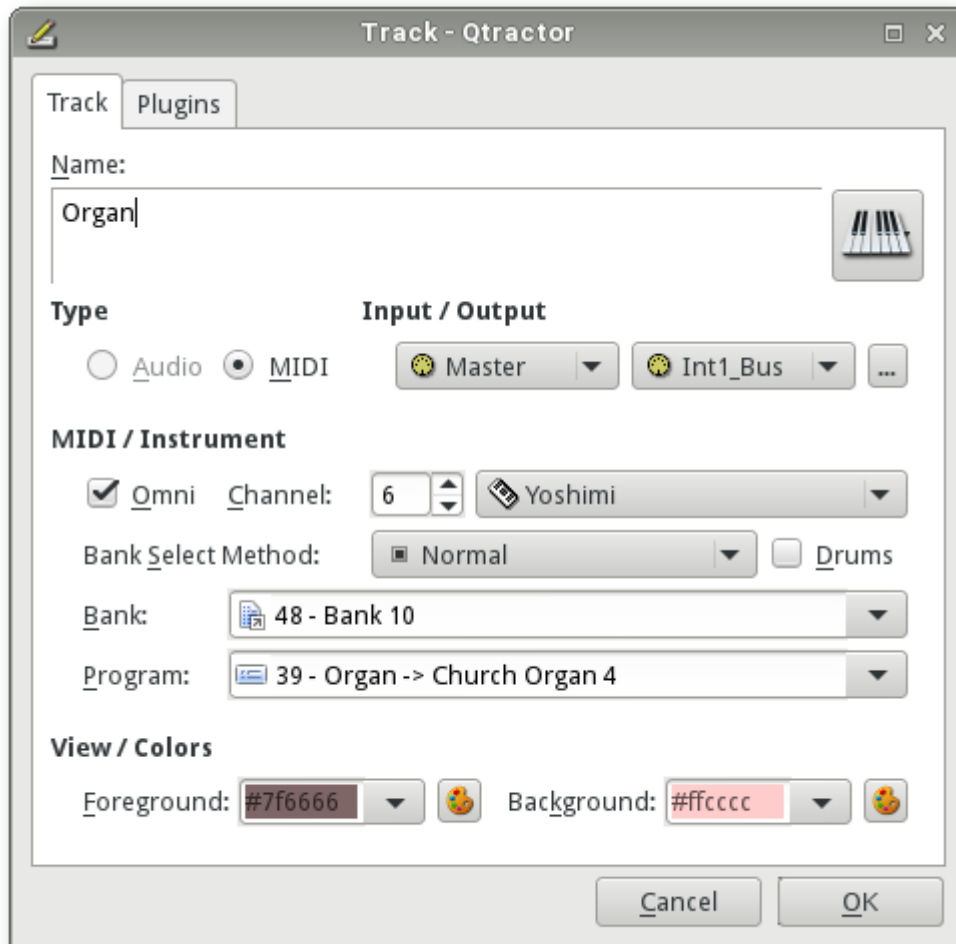
The tracks area has two panes: the left one displays the list of tracks with their respective properties and the middle pane is the audio and MIDI clips area, where multi-track composition and arranging is performed. Column widths in the left pane can be adjusted to allow for long track names etc. by dragging their edges; double-clicking resets them to their defaults. Track height can be adjusted by dragging up/down in the **Nr** column of the left pane, or by right-clicking and using the **Height** menu.



Audio and MIDI tracks with meters enabled

Note that information regarding global UI settings, such as column widths and Mixer set-up, is stored in `~/.config/rncbc.org/Qtractor.conf` (and so restored upon re-loading Qtractor), while track height is stored in the session file (and so restored upon re-loading a session).

To create a new track either right-click in the left pane and choose **Add Track**, or right-click in the right pane and choose **Track / Add Track**. In the **Track** window which appears, choose a name in the **Name:** box, choose a **Type** (*Audio* or *MIDI*), adjust any other options desired, such as the track's colour and icon, and whether it is for drums (which will show MIDI Note On events as diamonds instead of rectangles) or not, then click **OK**.



Track window for a MIDI track

You can also duplicate an already-existing track by right-clicking the track in the left pane and choosing **Duplicate Track**.

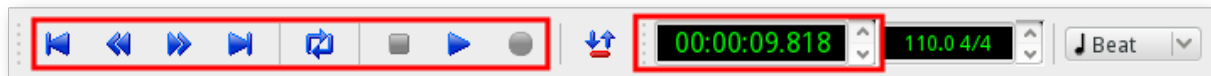
As is usual, tracks are stacked on horizontal strips and clips are layered on a bi-dimensional grid, in time sequence for each track strip. Time is modeled on the horizontal axis and pictured by a bar-beat scale ruler at the top of the track view. Clips may be conveniently aligned to discrete time positions, depending on the current *Snap* mode setting, which is set via either its icon on the toolbar or the **View / Snap** menu. When set to “None”, snapping is 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 is also available.

The red vertical line in the tracks area is the *playhead*. This shows the point at which playback/recording will begin when you start the transport rolling and its time corresponds to the clock towards the top-left of the screen. You can control the position of the playhead via the transport controls (to the left of the clock), by dragging the top of the playhead, or by **Ctrl/Shift**-clicking in the horizontal bar above the tracks which houses the bar numbers or in an area of the Tracks pane which contains no clip (**Ctrl/Shift**-clicking on a clip will select the clip). Alternatively, you can **Ctrl/Shift**-middle click on either the horizontal bar or anywhere in the Tracks pane (with or without a clip).



Transport controls (left box) and clock (right box) showing current position of playhead

The blue vertical lines are *Edit Markers*. See [below](#) for information on these.

4.6.2. Track States

A track's state is controlled via the **R**, **M**, **S** and **A** buttons in the left pane. Their meanings are as follows:

- **Record** - arm the track for recording, ready for creating new audio or MIDI clips with captured material
- **Mute** - silence the track so that it produces no sound
- **Solo** - set only this track to produce sound, silencing all the others (though those already solo-ed will too produce sound)
- **Automation** - automatically control various parameters of a track or its plugins. See the [Automation](#) section for details

When muting/solo-ing a track, holding down **Shift** or **Ctrl** while clicking the **M** or **S** buttons produces the following effects:

- **Shift**-click - Enable/disable state on all tracks. If the button clicked is currently disabled, all tracks will be enabled; if it is currently enabled, all tracks will be disabled
- **Ctrl**-click - Enable/disable state on this track only. For example, **Ctrl**-solo-ing a track with other tracks already solo-ed will un-solo those other tracks, leaving just the target track solo-ed

6	Bass	R M S A	Master/Yoshimi MIDI	*5	-
7	Saw	R M S A	Master/Yoshimi MIDI	*6	-
8	Drums_Cymbals	R M S A	Master/Fluid MIDI	*9	12
9	Bubbles	R M S	Master/Yoshimi MIDI	*7	-

Mute, Solo, Record & Automation buttons for individual tracks

It is possible to select more than one track at a time in order to mute/solo/monitor them all simultaneously, by using the following commands:

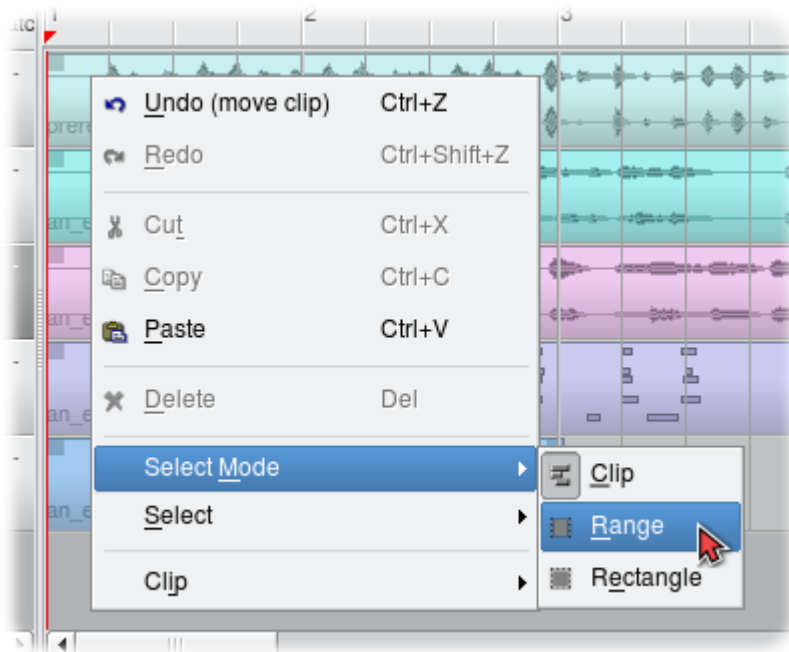
- Add tracks - **Shift**+left-click/mouse drag; **Shift**+Home/End keys; **Shift/Ctrl**+up/down arrow keys
- Toggle tracks - **Ctrl**+left-click/mouse drag; **Ctrl**+Home/End keys

In Qtractor parlance, though there is only ever one *current* track at any one time for editing purposes, there may be one or more *selected* tracks, invoked by the above commands, on which an operation can be performed. The *current* track is always displayed with a slightly darker background, whilst the colour of *selected* tracks is determined by the current palette scheme or theme (the default being that of the desktop environment). Once you have your desired tracks selected, you can mute/solo/monitor them using the relevant icons, menus or keyboard shortcuts. It is also possible to **Shift/Ctrl**-click the **M** or **S** buttons (as discussed above) with multiple tracks selected in order to apply these operations to the entire “set” of selected tracks.

Tracks can also be muted and solo-ed on mix-down, which will then apply when exporting, enabling you to export specific tracks only.

4.6.3. Making Selections: Select Modes

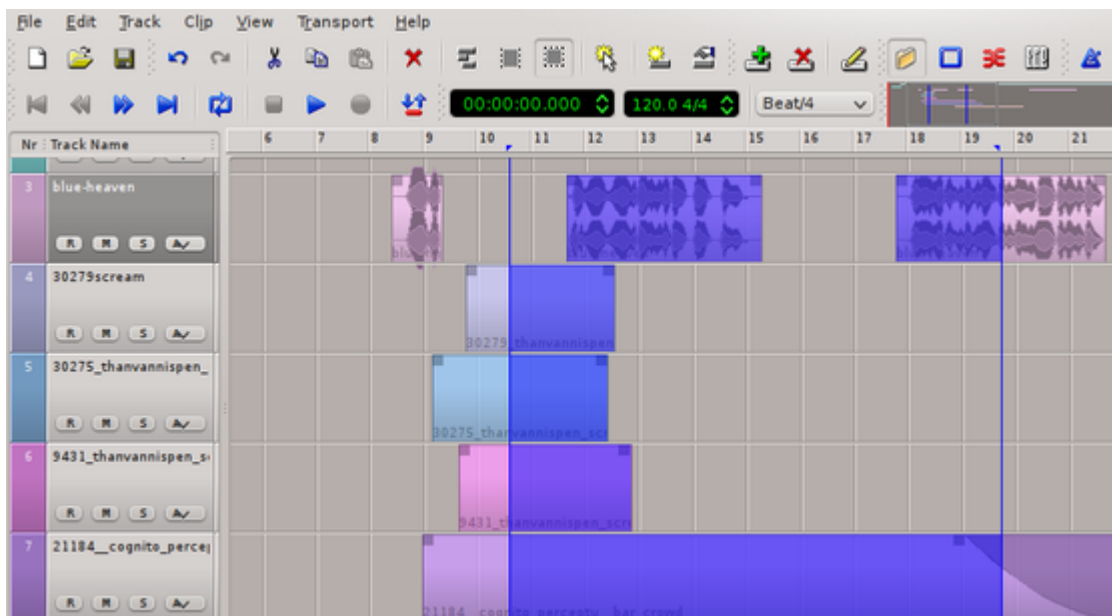
The mouse cursor has four *select modes*: *Clip*, *Range*, *Rectangle* and *Automation*. Choose your desired mode via **Edit / Select Mode** from the main toolbar, or by right-clicking in the track area and choosing **Select Mode**.



Select Mode context menu (right-click menu)

- **Clip** - allows you to select whole clips, click and drag them within and between tracks, and extend/shorten their lengths. The precision with which you perform these operations is determined by the current *Snap* mode setting
- **Range** - spans all tracks in your workspace and allows you arbitrarily to select any portion of clips, irrespective of their beginning/end. You can, for example, select a range in the very middle of a clip and all other clips above and below it, because you are selecting a block of time, rather than a particular clip
- **Rectangle** - is similar to *Range*, but is track-specific. This allows you to draw an arbitrary “box” area over one or more tracks, the extent of which becomes your selection

Once you’ve made your selection, you can then perform the usual cut, copy, paste etc. operations.

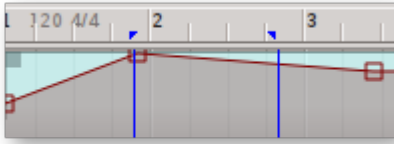


Making a selection using the Range* select mode*

For the final select mode, *Automation*, see [its section](#).

4.6.4. Making Selections: Edit Markers

In addition to using select modes, you can select a range in your workspace by using the *Edit-head* and *Edit-tail* markers. These are the two blue vertical lines which are present in the tracks area and which are independent of the playhead.



The Edit-head and Edit-tail markers

To set the *in-point* of your selection, left-click in the bar/tempo area of the workspace, where the top of the blue lines are situated. To set the *out-point*, right-click in the same area. You can also drag the lines. Middle-clicking will bring both lines together at that point.

Once you've set your markers, you can use them to make a selection. To do so, from the main toolbar, choose **Edit / Select** and one of **Track** or **Track Range**. You can also right-click in the track area to access the same menu.

4.6.5. Loops

Looping a section is a common operation, often used to enable a musician to practise a particular passage, or a mix engineer to focus on a specific area. To play a section of your composition in a loop:

1. Select a range, using one of the methods described above
2. From the **Transport** menu, select **Loop Set**. Alternatively, click the *Loop* button (pair of blue, circular arrows) on the main toolbar
3. Move the transport into the loop range, or somewhere before it, then press the *Play* button or the space bar. Once the transport reaches the end of the loop range, it will return to the in-point and seamlessly continue playing until stopped

It is also possible to record using a loop, but such an operation would more commonly be handled by Qtractor's *Takes* feature. See [its section](#) for details.

4.6.6. Paste Repeat

In electronic music especially, there's often a need to record one or two bars and then loop them to create standardised bass lines or drum beats. For drum tracks, you may prefer to handle them by using a specialist drum machine, such as [Hydrogen](#), which would play the same data (either pure MIDI data or audio samples) for as many bars as you program it to. However, the same effect can be achieved in Qtractor by using a *Paste Repeat*. This allows you to copy an audio or MIDI clip and then paste it back-to-back for as many repetitions as you wish.

There are two ways to perform a Paste Repeat - one quick and one slightly more detailed. The quick method is to **Ctrl**-click and drag a clip's left or right edge. This will replicate the clip as many times as fits into the desired area. The more detailed method is as follows:

1. Check your snapping settings in the **View / Snap** menu (when pasting clips for a loop, the user generally wants precision snapping to the clips' edges)
2. Select the clip you want to copy using the *Clip* select mode (or use *Range* mode, *Rectangle* mode, or the Edit Markers to select a portion of a clip) and use **Edit / Copy** to copy it
3. Click **Edit** and choose **Paste Repeat**
4. In the **Paste Repeat** window, enter the desired number of iterations in **Count** and, if needed, the duration (in frames, time or BBT) between the start of each iteration in **Period**. Click the **OK** button to confirm
5. Position the box outline where you want to paste the clips and left-click to anchor them in their track

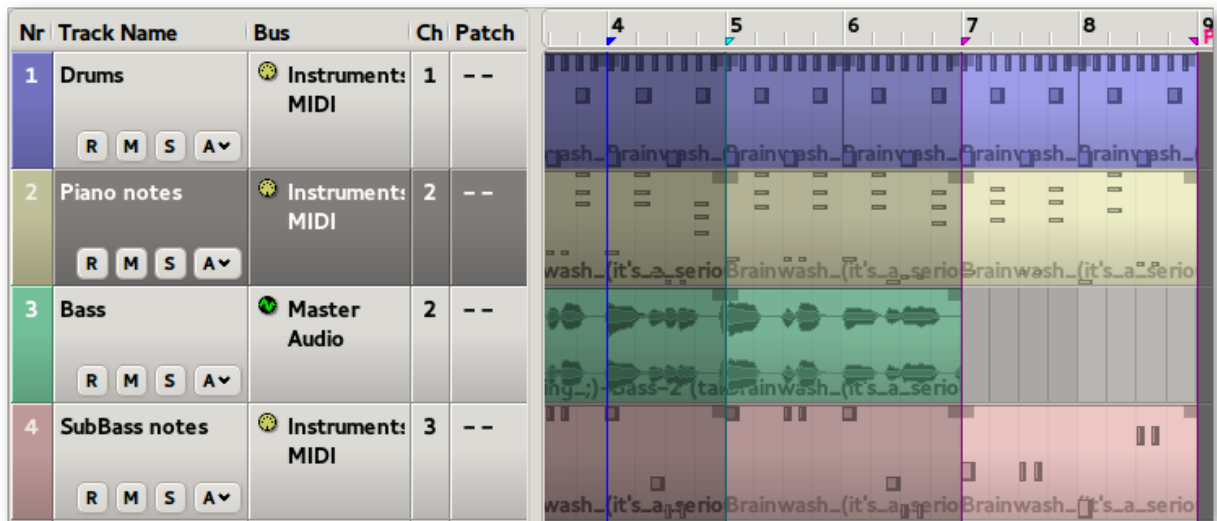
Note that any clips copied in this way will become linked to the original, as covered in the [Clips](#) section.

4.6.7. Punch In / Out

If one or two bars of a track don't quite measure up to your standards, you might want to re-record just those bars without having to record the entire track over again. This process is called *punching in/out*.

To perform a punch in/out:

- Use the *Range* or *Rectangle* selection mode, or the Edit Markers, to define the region of your timeline that you wish to re-record
- Once your range is set, click the *Punch in/out* button in the main toolbar (pair of up/down arrows), or use the **Transport** menu and select **Punch Set**
- Arm the recording destination track by clicking its "R" button in the track properties pane, on the left hand side of the main workspace
- Press the *Record* button in the main toolbar, position the playhead somewhere before your selected region, then press the *Play* button or the space bar. Qtractor will play back your piece and you can play along, but nothing will actually be recorded unless it is within the *punch* range
- To stop playback, click the *Stop* button in the main toolbar, or use the space bar



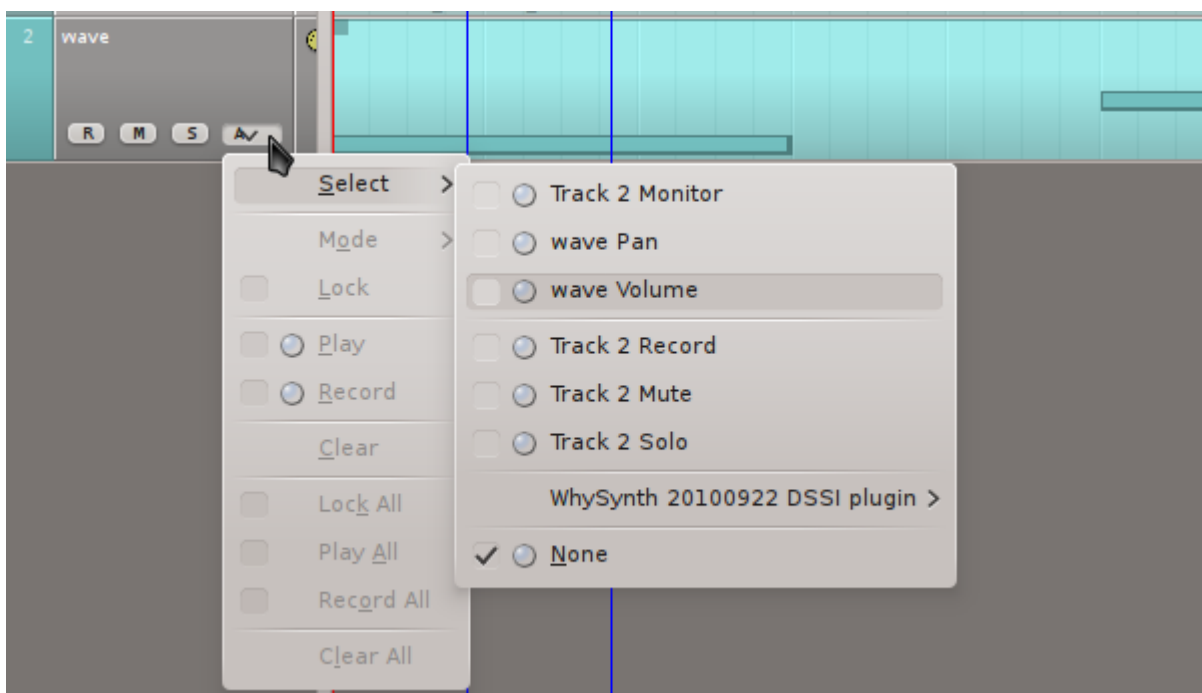
Punch in/out range (purple markers) inside a loop range (green markers)

4.6.8. Automation

Qtractor features automation of nearly any aspect of any plugin, as well as basic track functions like volume and panning.

Note: In principle, only tracks can be automated, not buses. However, there is a method to automate buses via tracks. See: [How To - 14 Automate Buses and Tracks by Layer with Insert Controller](#)

- In the track you wish to automate, click the *Automation* button in the track list. From the pop-up menu, select the attribute of the track that you wish to automate first (here, we're automating "wave Volume"). An overlay is placed on the track.



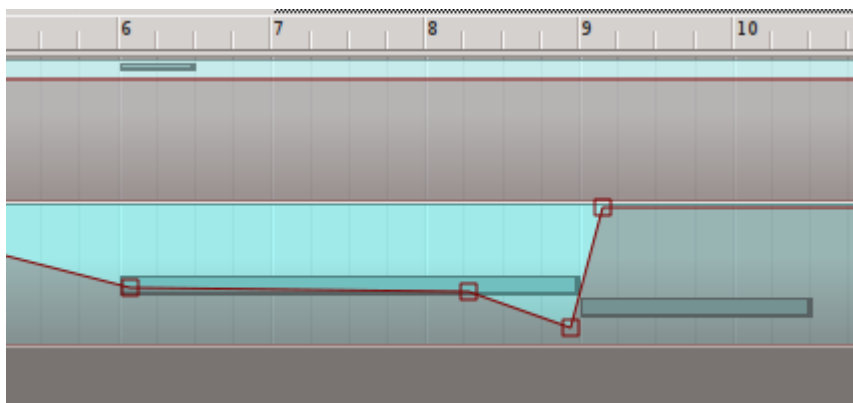
Automation pop-up menu

- Click the **Edit** menu to go into the **Select Mode** category, and choose the **Automation** selection tool. Alternately, click the *Automation Selection Tool* icon from the top menu bar
- Armed with your automation selection tool, click on the automation overlay on your track to create a node. Node placement is constrained by the *Snap/beat* setting, so set this to *None* if you wish to place a node at an arbitrary point
- Adjust the volume of this node to be the initial volume. You can either drag the node or double-click it and enter a precise value
- Click again on the automation overlay to create a second node, and move it to another volume level. For the space between the two nodes, the volume will change from the starting node level to the ending node level
- You can drag multiple nodes left/right, maintaining their values (vertical positions), by first selecting the nodes then dragging them whilst holding Ctrl or Shift. It is not currently possible to drag multiple nodes up/down



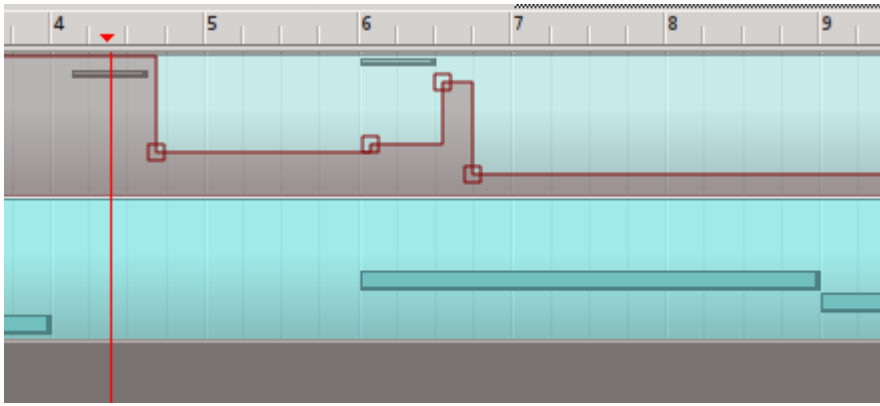
Adding automation nodes

If you need a less gradual change, move the nodes closer together, or use more nodes to shape different movements.



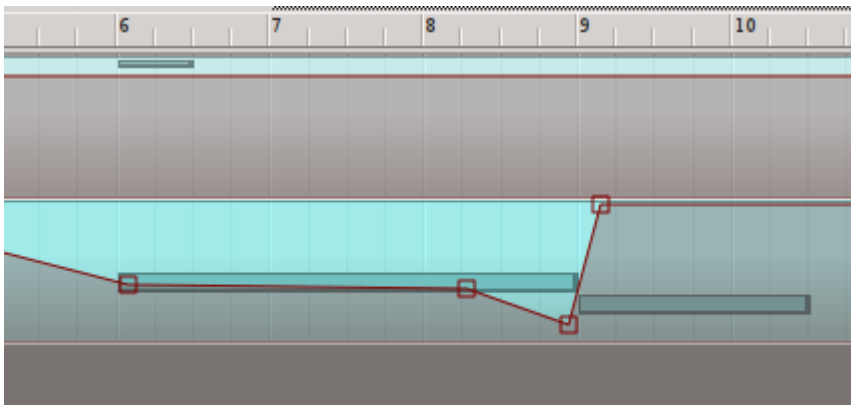
Adjusting automation nodes

The type of transition used between automation nodes can be customized via the track's **Automation** button, within the **Mode** category. **Hold** places the automation nodes into absolute values until otherwise changed.



“Hold” automation mode

- **Linear** - effects an even, linear progression from one node to the next. This is the default automation mode.



“Linear” automation mode

- **Spline** - same as Linear, except with B-splines instead of a completely linear progression between nodes.
- **Logarithmic** - same as Spline, except that the degree of the Bézier curves changes proportionately to the difference between nodes.



“Spline” automation mode

- **Color** - when you automate volume, panning and effects all on one track it can get confusing. Use **Color** to change the shade of the automation overlay so that you can distinguish one set of controls from another. **Lock** prevents accidental changes to automation; this is especially useful when working on a different automation function on the same track.

If you are a skilled live mixer and want to record your performance in real-time, you can override existing automation decisions, or record new ones, in two different ways, each available in the Automation menu:

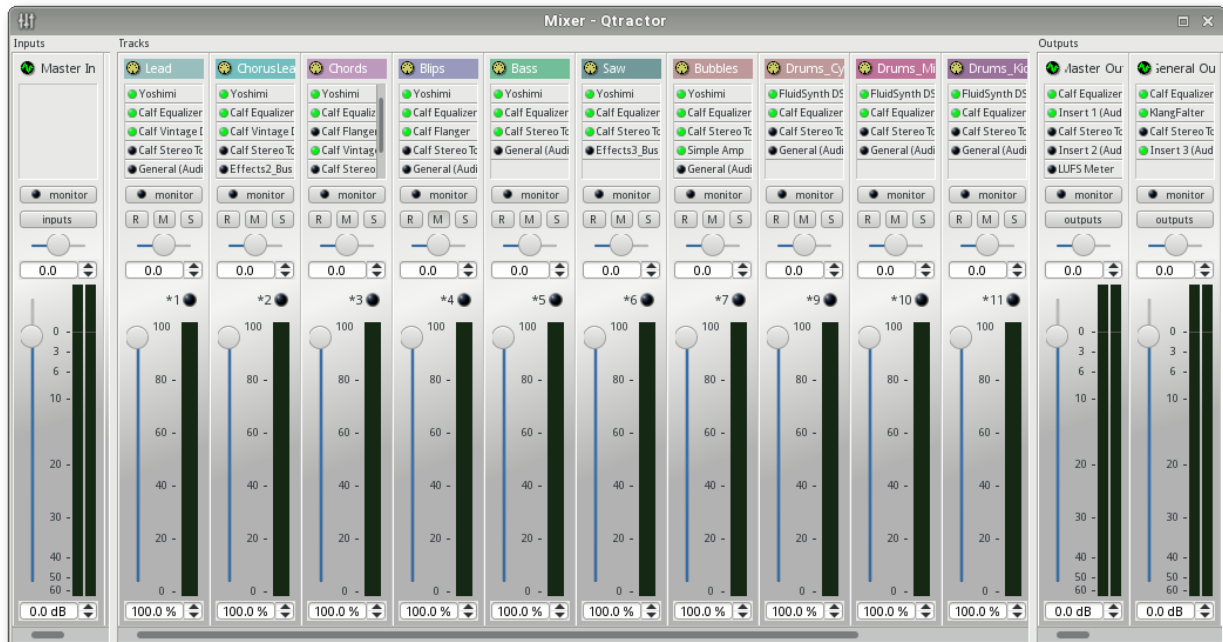
- **Play** - with this enabled, you may make real-time changes to existing automation, and upon release of the slider or knob that you are automating, the value returns to the pre-existing value. This may be combined with the Record option.
- **Record** - with this enabled, you may make real-time decisions for automation and every move will be recorded, in real-time, as a node in your track. They can be manually changed later, if needed.

4.7. Mixer

4.7.1. Mixer Summary

The Mixer window serves for session control, monitoring, recording and assistance in mix-down operations. It is divided into three panes: the left accommodates all input buses; the centre, individual track strips (either MIDI, with a yellow “plug” icon, or audio, with a green “waveform” icon); and the right, output buses. These are marked as Inputs, Tracks and Outputs respectively. Right-clicking in this area allows you to toggle the Inputs/Outputs’ displays on and off; you can also left-click and drag & drop them both within and outside of the Mixer.

The Mixer window can be accessed via the **View / Windows / Mixer** menu. It opens as a separate window, making it a prime candidate for occupying a second screen if you have a multiple-head set-up.



The Mixer window

Each vertical mixer strip offers a volume slider (MIDI tracks) or gain control (audio tracks and buses), panning slider, Mute/Solo/Record state buttons (tracks only), monitor button, connections button (buses only) and output event activity LED (MIDI tracks only).

Right-clicking in the panel directly underneath each strip's name allows you to perform a variety of operations. For instance, you can add [plugins](#), or set up dedicated outputs via **Audio / Dedicated**. A dedicated output is an audio output specifically for that track, which can be connected via JACK (in Qtractor's **Connections** window, or QJackCtl, it will appear in the **Audio** tab, under *Qtractor*). You can also choose whether to **Auto-connect** the dedicated output via the same right-click menu - doing so means that it will automatically be connected to *system:playback_1* and 2.

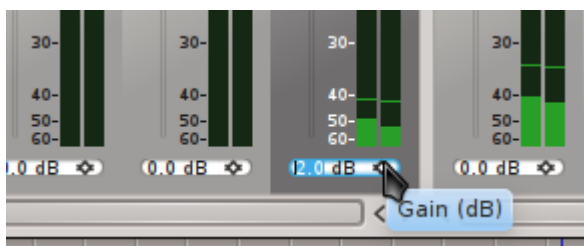
At the extremities of the Mixer window are the *Master In* and *Master Out* buses. These are independent of any one track and control everything coming into or going out of the Mixer. Here, you can adjust the master input/output levels and place any effects processors you want to apply globally (such as a compressor to prevent peaks on outgoing audio).

Monitoring is presented in the form of peak level meters for audio and note event velocity for MIDI, both with fall-off eye-candy. 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 the respective channel's Controller 7 and system-exclusive master volume for output buses. MIDI pan control is only available for track strips and is implemented through the channel's Controller 10. MIDI input buses have volume and pan controls disabled.

4.7.2. Setting Levels

When monitoring the levels of your tracks, make sure that the sound does not enter the red zone on the volumeter. In audio production, any sound that goes above 0dB hits its peak and can cause distortion. It's not uncommon to flirt with 0dB to truly maximize the impact of important sounds, but leaving sounds above 0dB in your final mix can ruin your project.

Using the volume level slider, adjust your track volumes to safe settings while listening to your project. Then, listen to your project again to refine the levels in relation to one another. If the sound source is too low (or too loud), you can add (or subtract) gain at the input point with the Gain adjustment at the very bottom of the Mixer window. This adds or subtracts gain level at the point of the sound's entry, meaning that you are changing the amount of sound reaching any effects processors. Generally, gain adjustment is avoided and other tools, like a Compressor or Equalizer, are favoured.



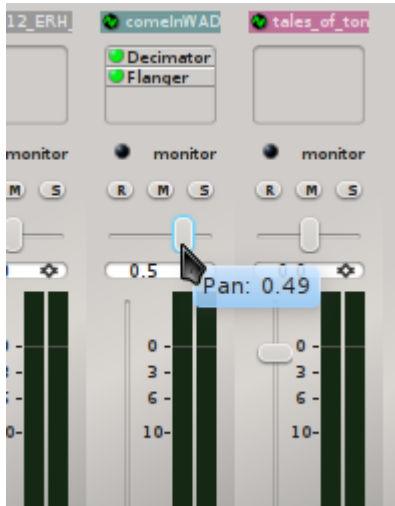
The Mixer's gain adjuster

Remember that Track levels don't exist in a vacuum; the levels are cumulative. So, if you have two tracks with near-maximum levels that produce very loud sounds at the same time, the *Master Out* level will **peak** even if each track was "safe" independently.

There is no authoritative way of achieving a balanced mix. Some professional mix engineers favour an additive mix, in which they start all levels at a reasonable and safe level and then accentuate important parts by adding volume to tracks. Others favour a subtractive method, starting everything at the highest safe (or unsafe) level and then bringing volumes down so that everything falls into place. Predictably, still others prefer a combination of these methods.

4.7.3. Panning

By default, tracks are set to play from the center point of the stereo space, meaning that the sound will feel as if it's situated directly in front of the listener, or spaced evenly between his/her ears. To diversify the sound, perhaps in order to emulate a live set-up, to suggest that an instrument is just next to the listener, or simply to broaden the scope and impact of the piece, you can *pan* tracks between the left and right speakers using the panning slider.



The Mixer's panning slider

When panning a track, “less” is often “more” and an even spread is usually best. In other words, instead of arbitrarily assigning tracks in stereo space just to give a spread to your sound, try to consider their overall balance. Avoid moving sounds to the extreme stereo positions (also known as *hard left/right*), since, in real life, sounds are rarely only heard in one ear and not the other. The sound of extreme stereo separation was common in early stereo mixes from the 1960s, in which the drums might only be heard in the right ear and the guitar only in the left; the effect was revolutionary at the time but is largely considered utterly unnatural now.

Generally, important tracks (vocals, lead etc.) and low frequency/groove tracks (bass, kick drum etc.) are positioned in the middle, while other parts are panned to the sides in various degrees. The extreme edges are often used for atmosphere, emphasis, or effect.

4.8. Signal Flow

4.8.1. Signal Flow Summary

For anything other than the simplest of projects, you will likely want to manipulate the flow of MIDI and audio data, routing it into, out of and through Qtractor in a variety of ways. You may want to drive an external sound module, accept input from a MIDI controller to record performance data, group together similar tracks to share the same plugin, or perform some other kind of processing before final export. In Qtractor, **Connections**, **Buses**, **Aux Sends** and **Inserts** are all used to control “what goes where”. There are many different ways to approach the process of routing and exactly what you’ll need will depend on your set-up. With this in mind, this section will give a general overview of the options available so that you can decide what best fits your requirements.

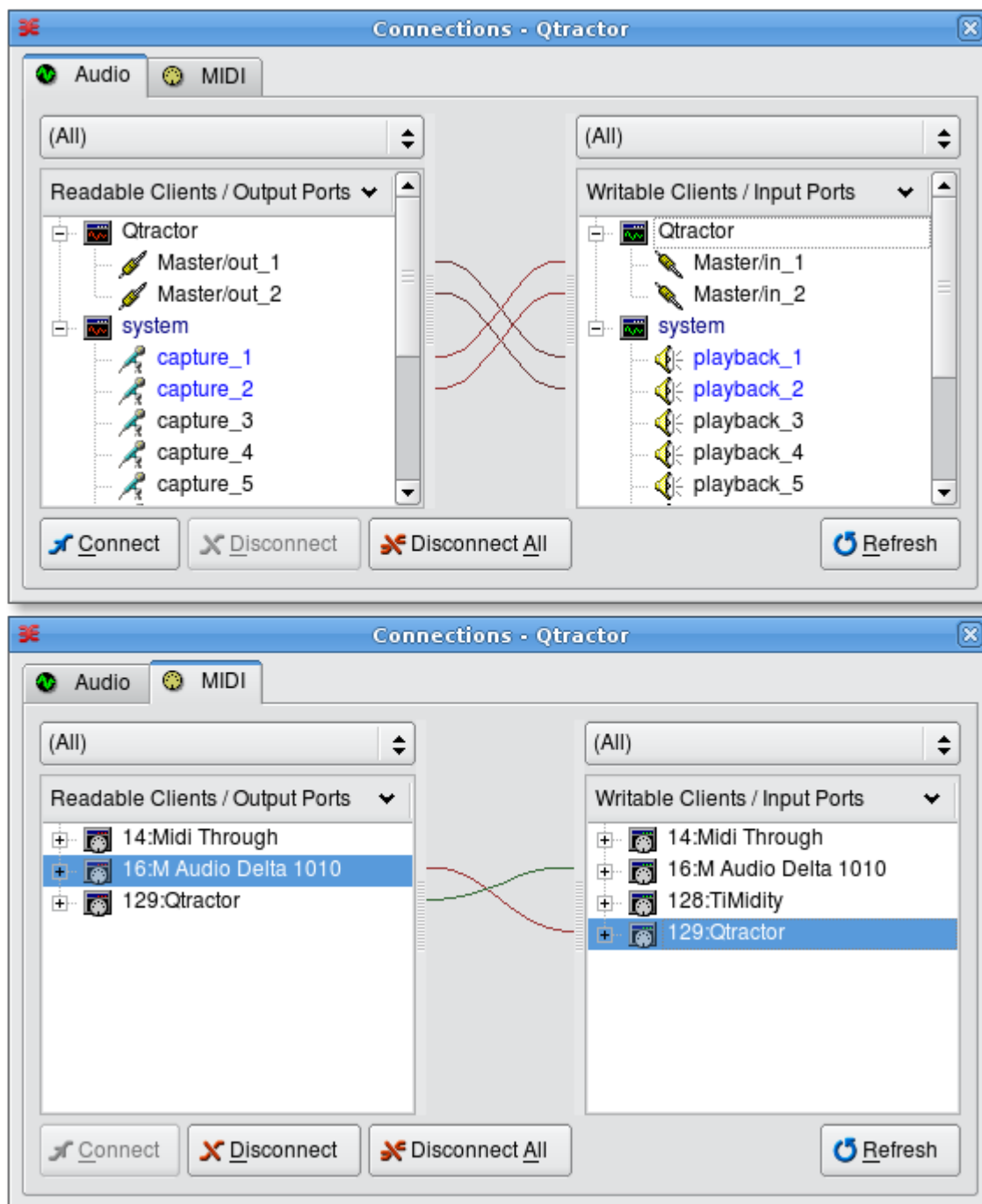
4.8.2. Connections

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. It can also be used to connect external client application ports - either JACK clients for audio or ALSA sequencer clients for MIDI. All active connections on the existing input and output buses are saved and restored upon session recall. To open the Connections window, select **View / Windows / Connections**, or click on its icon.



Connections icon

A window similar to the below will then open. Its features and usage are almost identical to those of [QjackCtl](#).



Qtractor's Connections window, showing both the Audio and MIDI tabs. "Readable" ports are sources of data (where audio or MIDI data can come from) while "writable" ports are places that data can be routed to (sent to)

Once you've decided how to approach the routing for a particular project, you can then make the necessary connections between Qtractor and whatever other programs or external devices you're using. You can connect a Readable/Output port (left pane) to a Writable/Input port (right pane) in several ways. One method is to click a port in the Output pane and, with the mouse button held down, drag the cursor to a port in the Input pane until the port is highlighted, then release the mouse button. Another method is to left-click a port in the Output pane, left-click a port in the Input pane, then either left-click the **Connect** button or right-click elsewhere in the window and select *Connect* from the pop-up menu. With this method you can also **Ctrl**-click or **Shift**-click to select multiple ports. Regardless of the method you use, a line representing a "virtual cable" will appear between the connected ports in the middle section of the window.

The *System* input ports *playback_1/2* represent your default left/right stereo output (e.g. your first-configured sound card). If you wish to use alternative destinations for playback, monitoring etc., you can connect them up here.

4.8.3. Buses

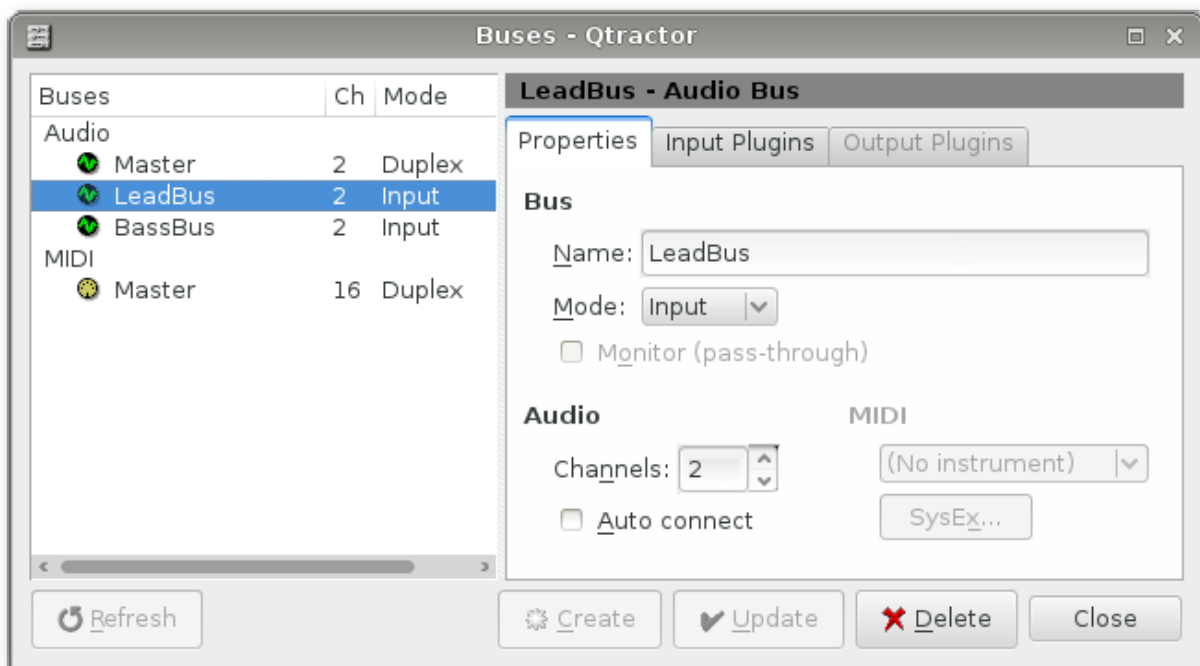
As explained at the start of this chapter, a bus is like a pipeline through which streams of data can travel. Qtractor has **Input**, **Output** and **Duplex** buses for both audio and MIDI. Input buses receive data streams from particular sources (inside Qtractor, or from other software or hardware), Output buses send streams to other destinations and Duplex buses both receive and send streams.

How many and what kind of buses you need will depend on what you want to achieve, but the basic steps to follow are:

1. Create a bus

- Assign it as an Input or Output for a track
- Connect it via the Connections window

To open the Buses window, click on the **View / Buses** menu.



The Buses window, showing one MIDI and three audio buses

The left pane contains a list of the session's buses; clicking on one of these will display its information in the right pane. The right pane has three tabs:

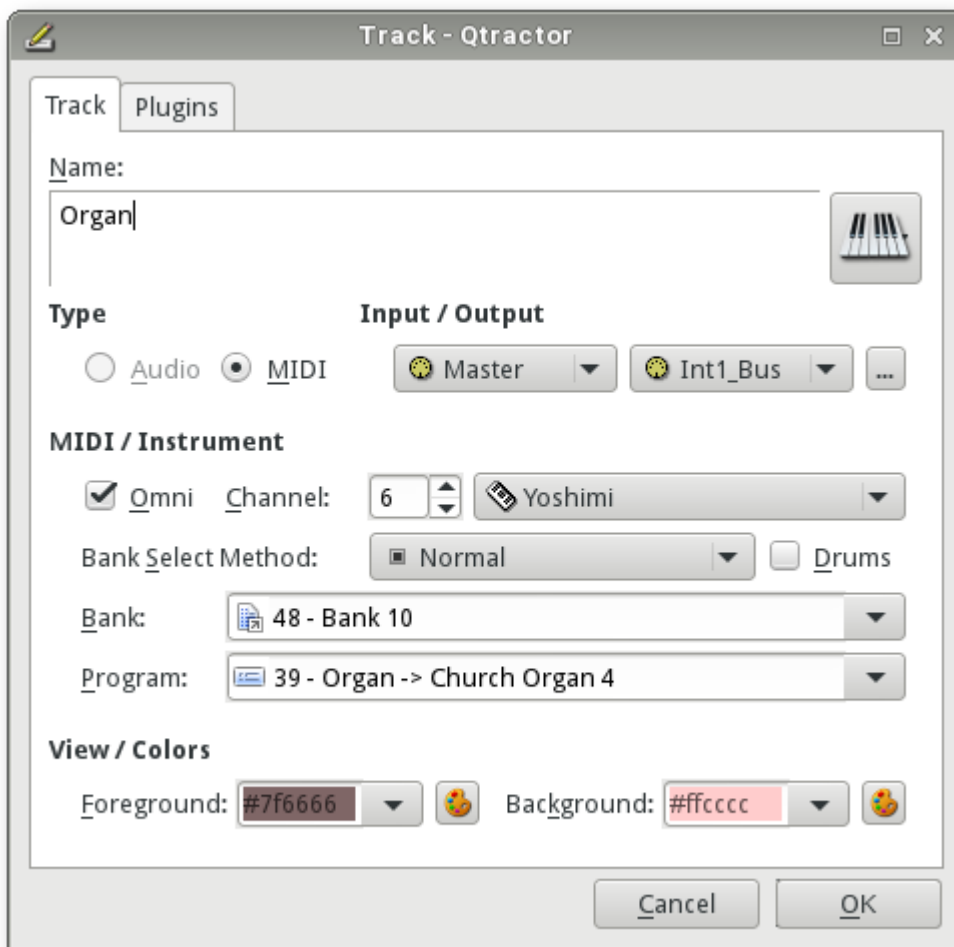
- **Properties** - Name, Mode (Input, Output or Duplex), Monitor (Duplex buses only), Channels and Auto-connect (Audio buses only) and Instrument and SysEx (MIDI buses only)
- **Input Plugins** - plugins processed on input (Input and Duplex buses only)
- **Output Plugins** - plugins processed on output (Output and Duplex buses only)

The method of creating a bus may not be immediately apparent. To create a new MIDI or audio bus, select any bus of the desired type in the left pane (if you're working on a clean session, the only choices will be MIDI and Audio Master) in order to display its properties in the right pane. Next, change the text in the **Name** field; this will enable the **Create** and **Update** buttons. To create a new bus, click **Create**.

Before clicking **Close**, you may wish to change other options in the right pane. For audio buses, a commonly used option is the **Channels** setting. Its default is **2**, which, for an Input bus, will give you a stereo recording. However, you may want to set this to **1** to create a mono input bus in order to record, for example, an external source via a single microphone. With a bus' **Channels** set to **1** and the bus assigned as the Input to a track, any recording on that track will then be in mono.

If you've made any changes, click **Update**, then **Close**. (If you don't click **Update** a reminder dialogue box will appear in any case.)

Once you've created your bus, you'll need to assign it to the relevant track(s). To do so, select a track and choose the **Track / Track Properties...** menu, or double-click the track in the left pane of the main view. This will open the **Track** window. In the **Input / Output** area, assign your bus via the drop-down box (whether it appears in Input, Output or both will depend on what type of bus it is).



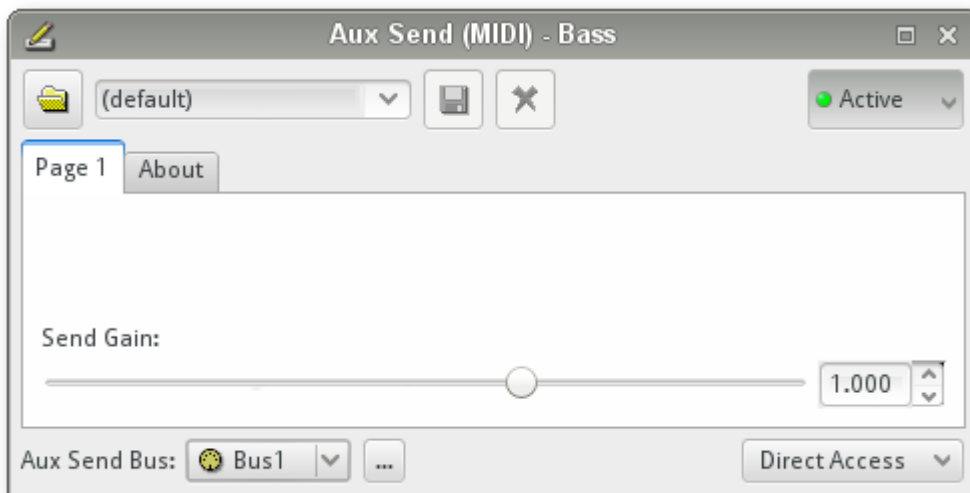
Track window for a MIDI track

The final step is to connect the bus up to its destination - a task which is achieved via the [Connections window](#). Again, exactly what you need to do will depend on your set-up, but if you wanted, for example, to send a particular MIDI track to an external software or hardware synth, you would connect its output bus' Output connection in the left pane to your synth's Input in the right pane. One thing you should *not* do is route the output of one Qtractor bus to the input of another, as this will give unpredictable results.

Buses are an extremely valuable tool and their use is almost essential for any reasonably sized project and when using Qtractor with external inputs and outputs. They enable you to use the full 16 MIDI channels through each bus, giving you much more flexibility than if you were using the Master MIDI bus only. You can, for example, have a dedicated MIDI bus routed to a multi-instrument synthesizer, sound module or soft synth, giving you 16 separate channels to control different instruments within the synth. Buses can also be used to route the audio output of external synths into Qtractor audio tracks for recording. For more information on this topic, see this [How To][How To - Sample MIDI Composition Workflow].

4.8.4. Aux Sends

Aux Sends are used to route audio or MIDI from a track or input bus to another destination *within* Qtractor - specifically, to a bus. To create one, open the Mixer, right-click under the name of your track or bus, then choose **Inserts / Audio** or **MIDI / Add Aux Send**. In the window which appears, set the Aux Send to "Active" (top right button), choose your destination bus in the **Aux Send Bus** combo box (bottom left) and choose the amount you want to send via the **Send Gain** slider.



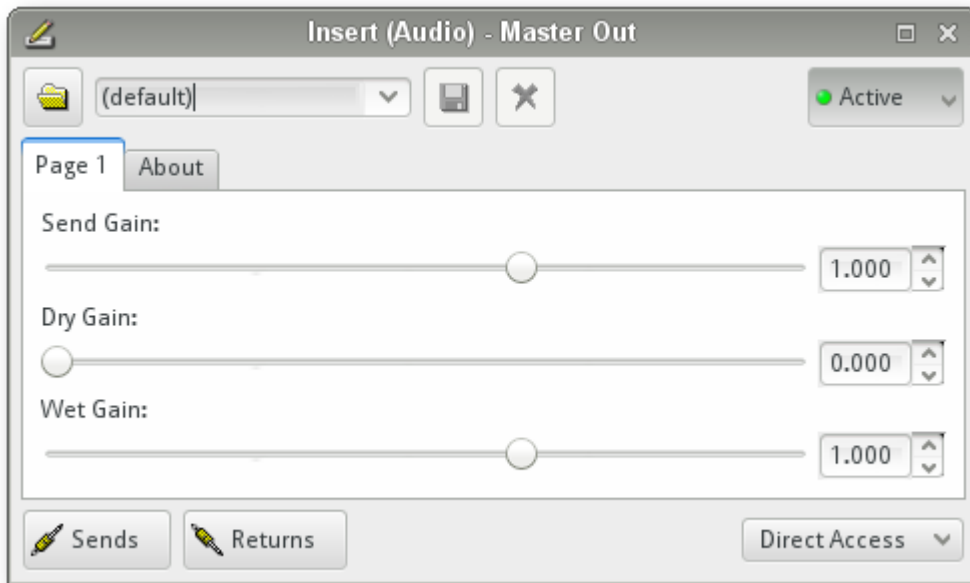
The Aux Send window for a MIDI track

What value to set **Send Gain** to will depend on the purpose of your destination bus. For example, you may want to send several tracks to the same bus in order to share that bus' plugin(s). This is a reasonably common requirement, particularly for reverb, where you would have an "overall" setting in the destination bus' plugin, then adjust the amount you send to this via each track's Send Gain slider. In this way, you can control the amount of reverb applied, in order, for example, to mimic an instrument's placement on the stage.

4.8.5. Inserts

Inserts are similar to Aux Sends, but are used to route audio or MIDI *externally* of Qtractor. They can be added to tracks, or to input or output buses. You can even route your entire mix to an external destination, such as [JAMin](#), before routing it back into Qtractor for export.

To create an Insert, first, launch your external application. Then, open Qtractor’s Mixer, right-click under the name of your track or bus and choose **Inserts / Audio** or **MIDI / Add Insert**. In the window which appears, set the Insert to “Active” (top right button) and adjust **Send Gain**, **Dry Gain** and **Wet Gain** as necessary.



The Insert window for an audio track

Next, use the **Sends** button to connect the Insert to your external application and the **Returns** button to connect the application back to the Insert (these buttons open up the same **Connections** window discussed above). The signal will now flow from your track/bus out to the external application, then return to the same track/bus and continue on from the point below the Insert.

4.8.6. Connection Persistence

When working with purely internal sources and destinations (plugin soft synths, Aux Sends etc.) Qtractor will automatically restore any connections you have made upon re-loading the session. With external applications, however, things are slightly different: Qtractor will only restore such connections *if they are present when the session is saved*. For example, if you save and close your session with everything correctly connected, upon restarting both Qtractor and your external application, the connections will be correctly restored. However, if you close the external application, then make a change in Qtractor and save (i.e. with Qtractor and the application in a “non-connected” state), upon restarting, the connection will be lost and you’ll need to make it again.

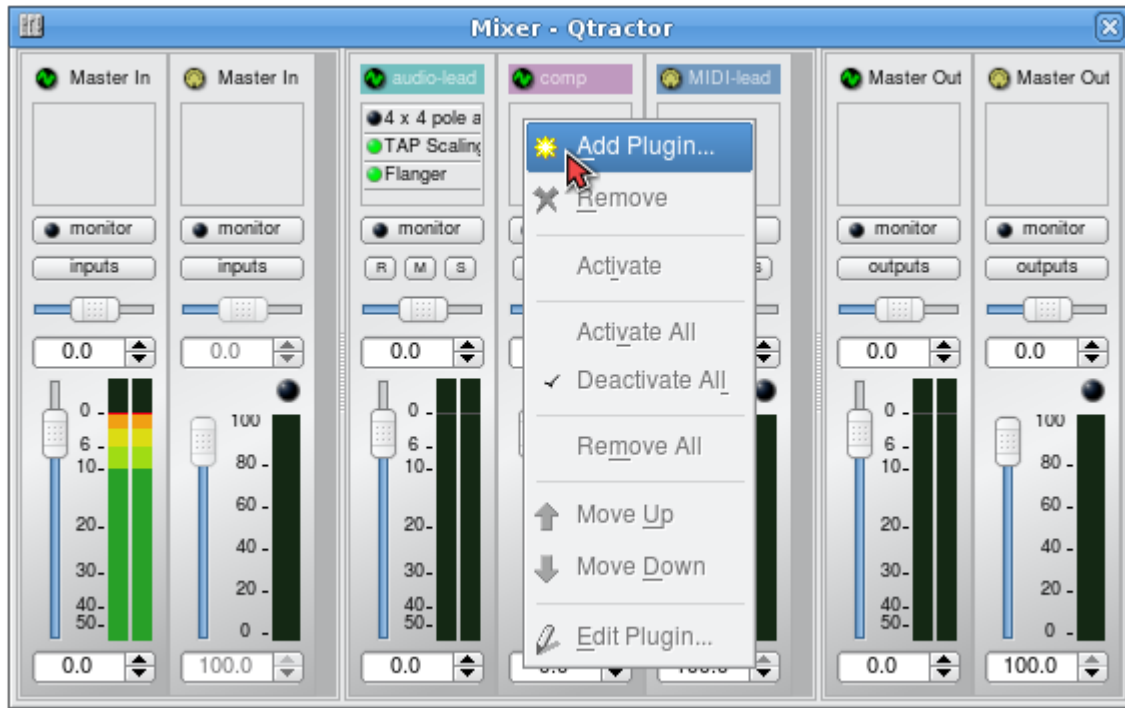
This behaviour can be avoided by always ensuring that you have your external connections in order before closing Qtractor (easier said than done when using many external applications). It can also be mitigated somewhat by using QjackCtl’s *Patchbay* feature to make the connections persistent; however, this too is rather unwieldy as it means adding each item manually to the

Patchbay. A third approach would be to use a session manager, such as QJackCtl's *Session* feature (which employs "JACK session"), or [NSM](#).

4.9. Plugins

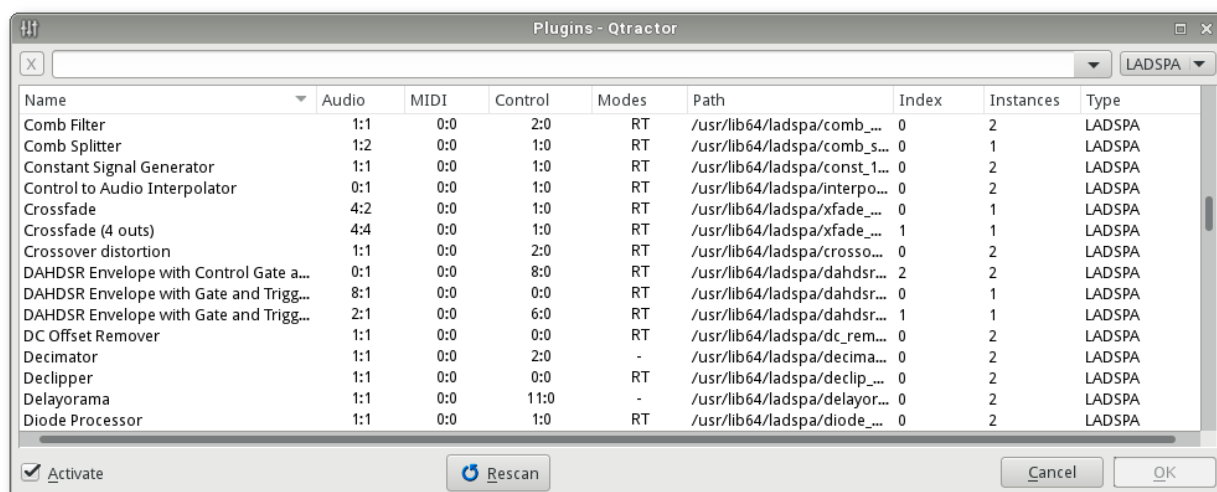
4.9.1. Plugins Summary

There are four types of plugins supported within Qtractor: LADSPA, DSSI, VST, CLAP and LV2. They can be added to both buses and tracks and are aggregated seamlessly as one single instance on a multi-channel context. To add a plugin, open the [Mixer](#), right-click in the panel below the name of the desired track and select **Add Plugin**.



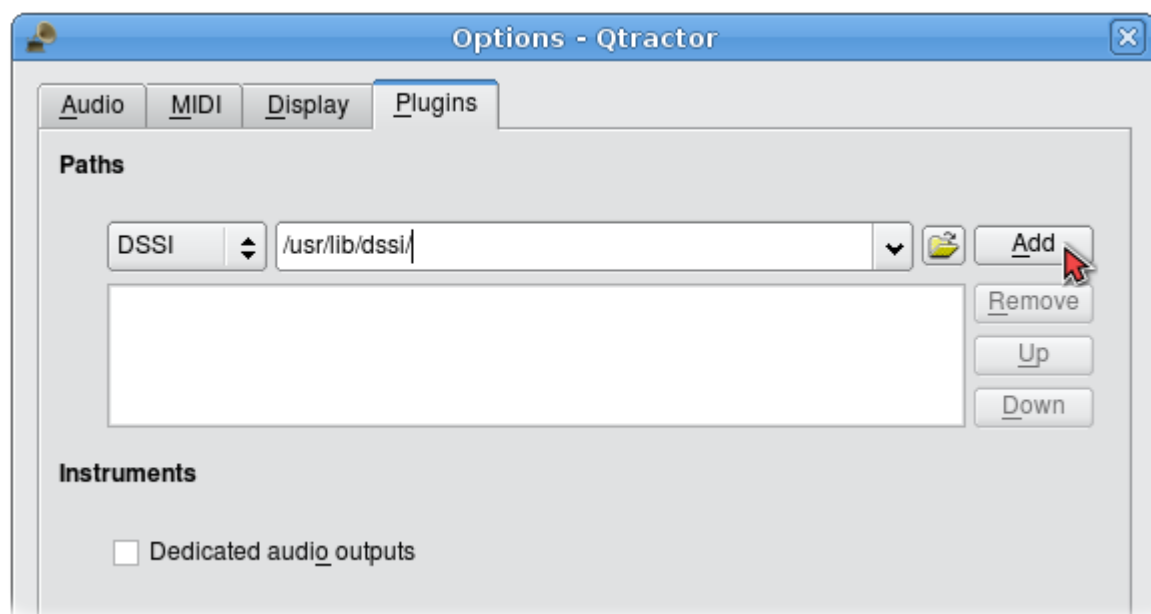
Mixer window showing plugin area at top. Plugins are added, removed, activated/deactivated, moved in the processing chain and their settings edited using a right-click context menu

In the **Plugins** window which then appears, select the plugin type - (**Any**), **LADSPA**, **DSSI**, **VST**, **CLAP** or **LV2** (see the following sections for information on the different types) - in the top-right box. Next, select your plugin from among those installed and, if desired, tick **Activate** in the bottom-left corner to enable the plugin as soon as it's added to the mixer strip. If you have a long list of plugins to choose from, you can use the top-middle box to search for what you want.



Plugins window, showing LADSPA plugins currently installed

Qtractor will look in the most likely installation directories for plugins and display any found in the above window. If, however, you have plugins installed in non-standard directories, you should specify their paths in the **Plugins** tab of the **View / Options...** window. Once you've made your selection, click the **OK** button to add the plugin.



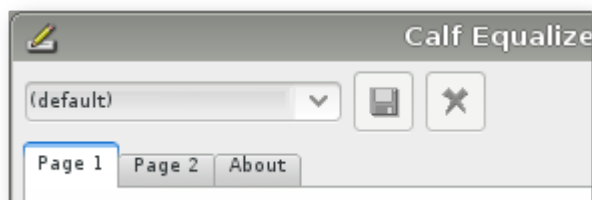
*Specifying the locations (paths) of plugins in the **View / Options...** window*

Plugins are processed in order from top to bottom and can be individually selected, activated and moved (by dragging/dropping) within the plugin chain order. They can also be copied or moved to other mixer strips by dragging/dropping and choosing your desired option from the pop-up menu which appears; a faster way to accomplish this is **Ctrl**+drag to copy and **Shift**+drag to move. There are a couple of things to note about plugins, which may affect your workflow.

- All plugins are *pre-fader* - that is, they are processed before the Mixer's Gain fader. If you want them to behave as though they were post-fader you can insert any plugin which controls gain at the start of the chain (i.e. before your pseudo post-fader plugins) and use that, rather than the Mixer fader, to control the track's gain

- There is no plugin delay compensation, so if a plugin adds latency to the signal chain this won't be corrected for. If you do need to correct for this, the only option is to do so manually by adjusting a clip's timing via the **Offset** value in its [Properties](#) window

Individual plugin control parameters can be modified in real-time through provided dialog windows and maintained as named presets for re-usability. To create a preset, once you have your desired settings ready, enter a name in the box at the top left of the generic plugin GUI (it will say “(default)” to begin with), then click the save icon next to it. You can select from among your saved presets by clicking the arrow next to the preset name or, when in the Mixer, by right-clicking the plugin and choosing **Preset**.



Generic plugin GUI, showing preset box and save and delete icons

Note that certain plugin-specific GUIs, such as [Calf LV2](#), may not display the presets information even though the functionality itself is there. In such cases, you can access the generic GUI by right-clicking the plugin in the Mixer and choosing **Properties...**, then create presets from there. When more than one GUI exists for a plugin, you can also opt to choose which you want by enabling *Select plugin's editor (GUI) if more than one are available* in the **View / Options...** / **Plugins** menu.

4.9.2. LADSPA

[LADSPA](#) has been a Linux audio plugin standard for many years. There are literally hundreds of LADSPA plugins available for Linux, providing a huge variety of processing effects, such as equalization, filtering, reverb, chorus, amp and speaker simulation and so on. Well-known LADSPA plugins include the [swh](#) package by Steve Harris and the [tap](#) package by Tom Szilagyi. These may already be present in your distro's repositories.

4.9.3. DSSI

[DSSI](#) plugin support is available for instrument and effects plugins. You must have the core DSSI subsystem installed in order for this type of plugin to function. Well-known DSSI plugins include the [fluidsynth](#) soft synth for working with soundfonts ([.sf2/.sf3](#)) and Sean Bolton's [WhySynth](#). Again, these may already be present in your distro's repositories.

4.9.4. VST (Linux Native)

Native [Linux VST](#) plugins are supported, but there are only a few native Linux VSTs available.

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 variable (the variable name is **VST_PATH**).

4.9.5. CLAP

Native [CLAP](#) plugins are also supported.

4.9.6. LV2

[LV2](#) is the extensible successor to LADSPA and offers many improvements over it. A wide variety of plugins are available, such as the [synthv1](#) synth, by Rui Nuno Capela (author of Qtractor), and the [Calf](#) set of effects. If you have the choice of using an LV2 plugin over a LADSPA or DSSI one, it's generally better to use the LV2 one. Doing so will ensure that when saving a Qtractor archive file (**.qtz**), all the relevant information is correctly included in the archive.

4.10. Working with MIDI

4.10.1. MIDI Summary

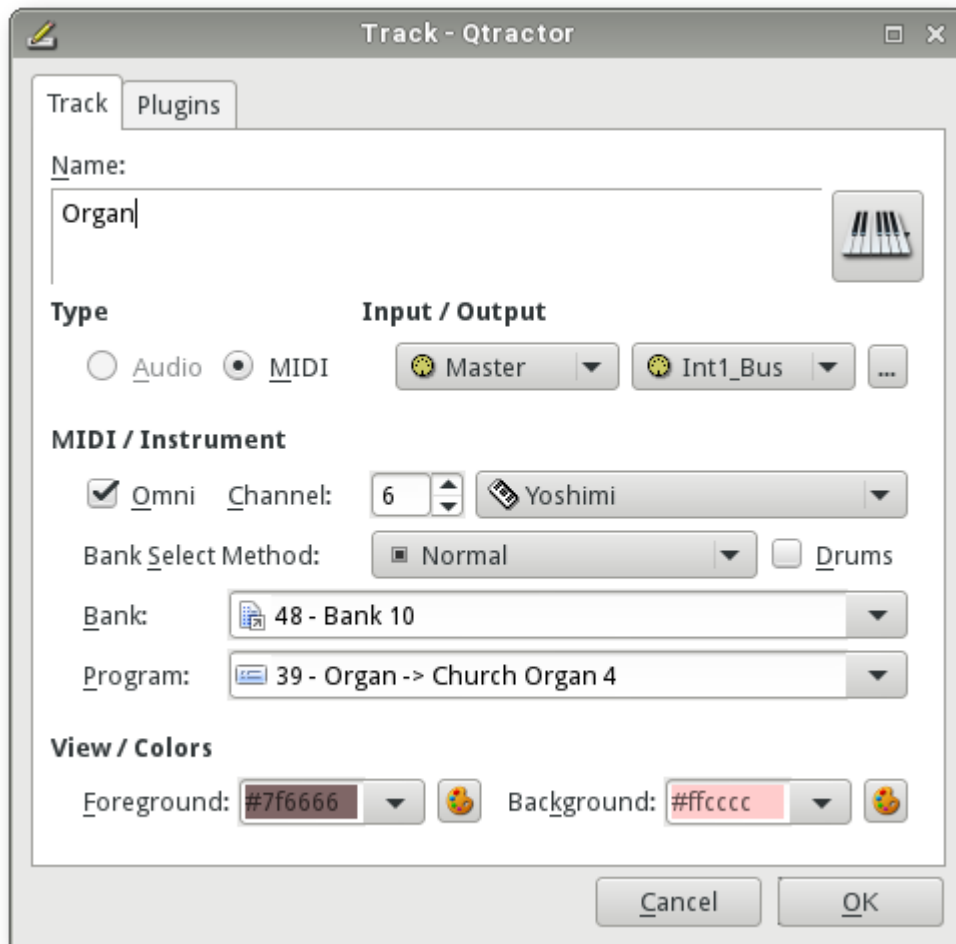
MIDI is a versatile format for triggering sounds. It can be used to trigger hardware synths external to your computer, software synth applications on your computer, samples, loops, automation functions and more. Some music programs bundle your MIDI data into your project file, which can make it difficult to share one MIDI loop between projects. However, Qtractor saves all MIDI tracks as independent files, meaning that you can easily re-use them.

There are basically three ways to get MIDI data into your Qtractor session and they are each covered in the sections below:

- Import MIDI data from an existing file
- Play in MIDI data via a USB MIDI keyboard controller
- Enter MIDI data into the MIDI Editor (sometimes called a “piano roll” or “matrix editor”)

Before beginning with MIDI, unless you're using Qtractor to drive an external hardware synth you should make sure that you have some [software synthesizers](#) installed on your system. If you've just installed soft synths now you should relaunch Qtractor so that it will detect the newly available plugins.

Note that with each successive MIDI [track](#) you add, you must increment the MIDI Channel used (Qtractor will do this by default), otherwise key presses on your MIDI controller, or playback from Qtractor, will use the same MIDI channel to trigger different sounds. In other words, if Track 1 and Track 2 are both using MIDI Channel 1, then a note played on Track 2 will also trigger the instrument assigned to Track 1. To change a track's Channel, double-click on the track in the left hand pane of the main view then, in the **Track** window which appears, select the **Channel** number underneath **MIDI / Instrument**. If you run out of channels, you can use multiple [buses](#).



Track window for a MIDI track, where the MIDI Channel can be selected

4.10.2. Importing MIDI Data

Since MIDI data consists of nothing more than digital signals to activate and deactivate sounds at certain times, MIDI files are small and easily distributed online. A simple internet search for MIDI files will result in thousands of files for nearly any song you can name. Taken alone, MIDI files are useless; imported into Qtractor and assigned instruments, however, the files come to life like a player piano. The MIDI file you import into Qtractor could be from the internet, one you created yourself in another application, or even a data dump from a MIDI-capable hardware synth. In this example, we will use one from the internet:

1. Download a MIDI file, such as [JS Bach “Invention in C minor, BWV 773 \(Canon\)”](#) and save it to your hard drive - preferably in your Qtractor’s session directory, to keep your project self-contained
2. In Qtractor, go to the **Track** menu and select **Import Tracks / MIDI**. Alternatively, you may open the **Files** pane, click on the **MIDI** tab and use the right-click menu to import the MIDI file without automatically adding it to a track (if you do this, you’ll need to drag the file into the workspace manually). *Please also see the note in the [Files](#) section regarding importing MIDI files*
3. The MIDI tracks will appear in your Qtractor session. Imported tracks will often have embedded General MIDI instrument assignments, as is the case with this example. General

MIDI was an effort to standardize a set of 128 instruments to provide similar playback results across all systems

If you have a soft synth installed and assigned to the relevant track(s) you can press the *Play* button in the top toolbar of Qtractor to hear the MIDI file. If you're using a General MIDI-compliant soft synth bank you should get a good approximation of how the original author of the file wanted the tracks heard.

4.10.3. Creating MIDI Data with a MIDI Controller

A MIDI controller is a piece of hardware to help you play the software-based synthesizers in your computer. Typically, it is a piano keyboard with a few octaves and no built-in sounds of its own. It usually plugs into the USB port of your computer and, once configured, will pass on any key press to Qtractor, allowing you to play a software synth in real-time as well as record the key presses themselves.

MIDI and USB are two technologies that, fortunately, have been kept universal enough that you almost don't need to question whether any given USB MIDI controller will work with your system. There are perfectly capable USB MIDI controllers being offered from vendors like Akai, Roland/Edirol, Digidesign/M-Audio and many others. If you require realistic, weighted keys with high resolution touch-sensitivity then look at the upper price-range of controllers and go to a music store to audition them. However, if all you require are a few octaves of piano keys so that you can get a tune into the computer quickly and easily, the basic controllers will be enough.

To use your USB controller to input MIDI data:

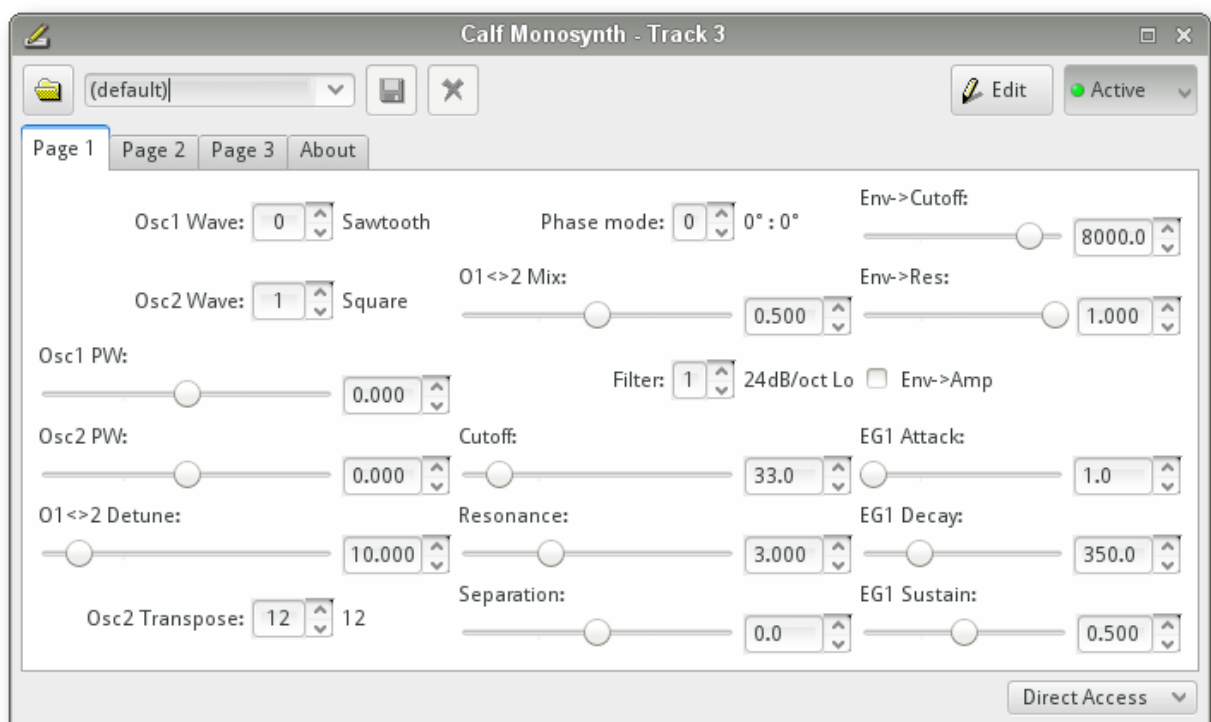
1. Plug the USB MIDI controller into your computer
2. Launch QJackCtl and start it
3. Launch Qtractor and click on the **View** menu, then select **Connections** from the **Windows** category
4. In the **Connections** window, click on the **MIDI** tab. Find your USB controller device from those listed on the *Readable Clients/Output Ports* column on the left hand side. Open its category to see the available outputs
5. Connect the output of your MIDI device to Qtractor in the *Writable Clients/Input Ports* column
6. Create a new [track](#), then double-click the track in the left pane of the main view. In the **Track** window which appears, make sure that the **Channel** value (underneath **MIDI / Instrument**) corresponds to the channel your MIDI device is transmitting on. Alternatively enable **Omni** alongside this to tell Qtractor to receive input on all channels
7. In the [Mixer](#), enable the track's **Monitor** button (so that it turns green)

*In Step 4, if you do not see your USB controller listed in the left column, check to ensure that your controller is powered on (some require external power; others are powered by the USB port) and check the cable connection. If you still cannot see the device in the left column make sure your computer sees the controller by checking `dmesg | tail`. If your computer registers the device but you are not seeing it in **Connections** click Refresh* in the bottom right of the window. If it still doesn't appear try restarting both QJackCtl and Qtractor.**

Presuming you have a soft synth installed and assigned to the track, pressing the keys on your USB controller should now trigger the soft synth in Qtractor, generating sound. To record MIDI data, arm your track by clicking its “R” button in the left pane of the main view, click the *Record* button in the top menu, then click the *Play* button to start the transport recording - every key press you make on your USB controller should now be recorded in real-time. Depending on the feature set of your controller, velocity, pitch and other data may also be recorded

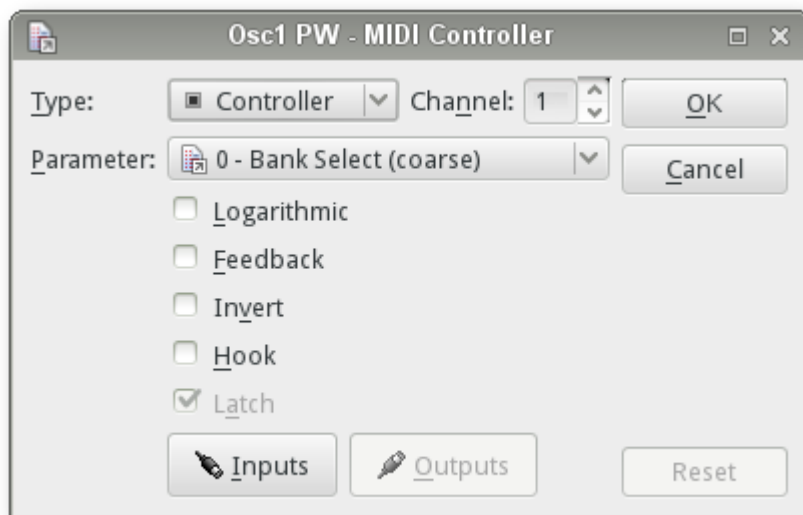
If you re-record over a clip, Qtractor will create a new, overlapping clip (which can, if desired, be merged with the original clip using **Clip / Merge**). If you’d rather *overdub* the original clip, adding extra notes directly to it instead of creating a separate clip, select the desired clip and choose **Clip / Record** from the main menu, or **File / Record** from the MIDI Editor. Once selected, the clip will change colour and any data recorded in its region of the timeline will be added to it.

You may wish to control particular settings of a plugin by sending MIDI controller messages to it from your USB controller. Although assigning MIDI controllers to LV2 plugins is not permitted by the LV2 specification, Qtractor offers another way of handling this which works for all plugin types. To assign a controller, first open the plugin’s generic GUI, by right-clicking on its name in the Mixer and selecting **Properties....** A window such as the below should appear, the content of which will be defined by the plugin’s exposed settings.



A plugin’s generic GUI

Next, right-click on the name of the setting you want to control and, in the dialogue box which appears, choose **MIDI Controller**. The below window should then appear.



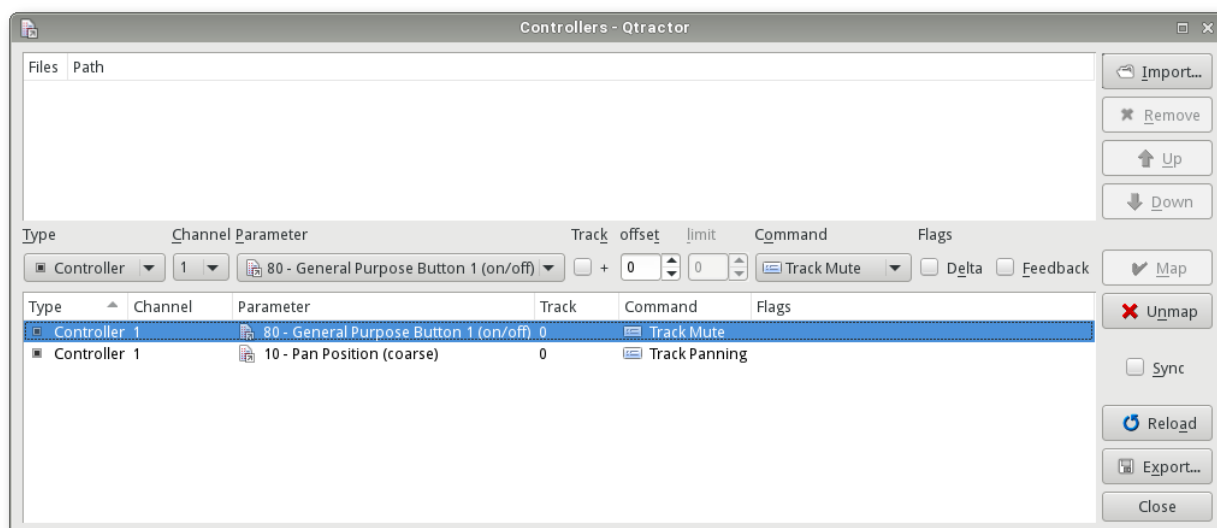
MIDI Controller window, where controller messages can be assigned to plugin settings

If you have not already done so, connect your hardware MIDI controller up, using the **Inputs** button. Then, choose which controller message to use via the **Type** and **Parameter** options, together with your desired MIDI channel in the **Channel** box. There are also several other options below these:

- **Logarithmic** - the linear scale of the MIDI controller (rotary knobs, sliders, etc.) is mapped to a logarithmic scale value on the target controlled subject. This is usually applicable to audio volume, dB scales or even frequency spectrum ranges
- **Feedback** - causes any change in the state value of the controlled subject to be sent/feedback to the MIDI controller, to allow the latter to reflect and sync its state accordingly. This is mostly applicable to MIDI controllers with visual (e.g. LEDs) and/or mechanical (e.g. motorized faders) feedback
- **Invert** - the minimum-maximum range of the MIDI controller is inversely mapped to the maximum-minimum scale values on the target controlled subject (and vice-versa)
- **Hook** - the MIDI controller and its controlled subject state/value are always in sync (hooked to each other); otherwise a deferred sync mode is in effect, with both ends playing catch-up to each other's from either direction until they match their states/values
- **Latch** - this is used only for on/off toggle parameters and relates to *Latch* vs *Momentary* modes of switches. These are the opposite of each other, so latch=OFF is the same as momentary=ON and vice-versa. This option is mostly applicable to foot-switches or pedal controllers. *Latch* (=ON) mode is the default setting for most such controllers, but a controller that operates in *Momentary* mode will usually send an event when you depress the switch (0 -> 1) and another when you release it (1 -> 0). (This is the meaning of "momentary": the switch state is only signalled to be momentarily on while the switch is depressed, returning to its former state upon release.) Setting the **Latch** option to OFF for this latter case will simulate a toggle switch, meaning that the target subject state will only change to ON or OFF when the switch is *either* being depressed *or* released, respectively

Note that MIDI controller assignments are applicable only to the session you create them in. If you want to use the same assignment in a different session you'll need to create it again.

You can also create and manage generic MIDI controller mappings, plus import/export these mappings as a **.qtc** file, via the **View / Controllers** menu.

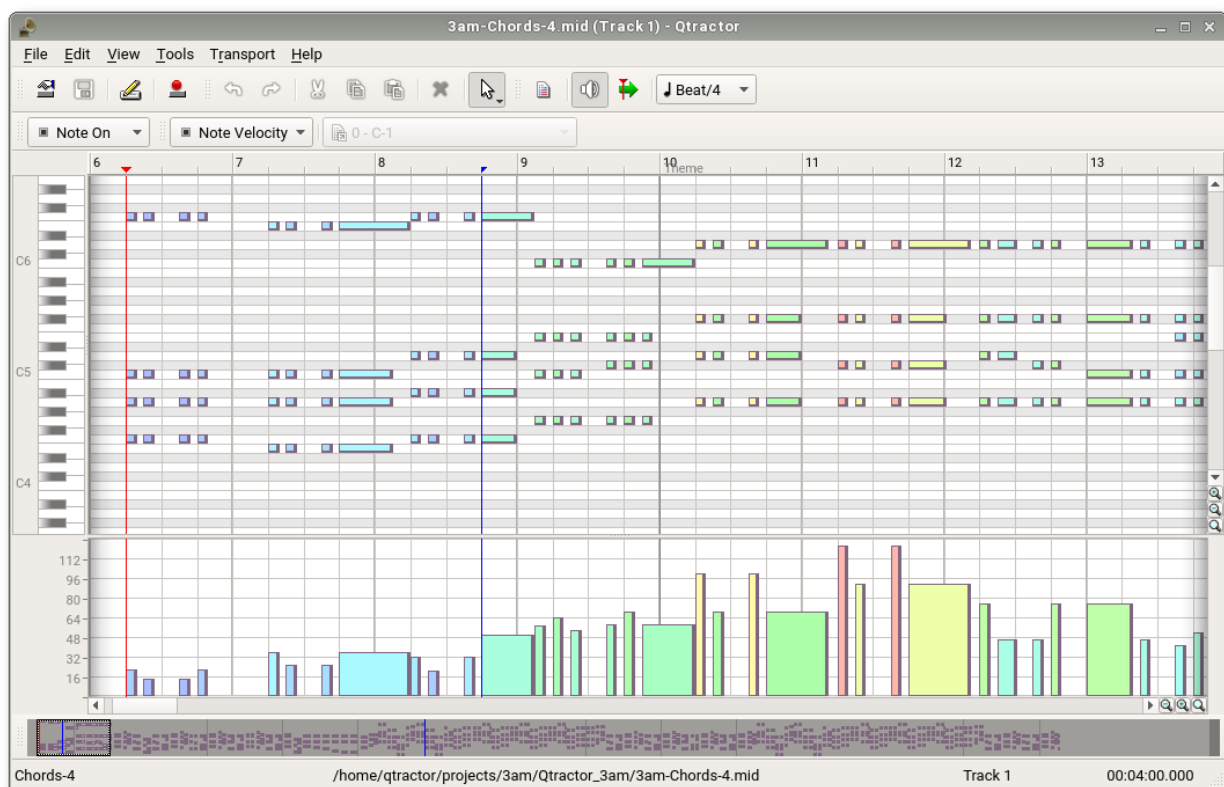


The Controllers window, where generic MIDI controller mappings can be managed

4.10.4. Creating MIDI Data with the MIDI Editor

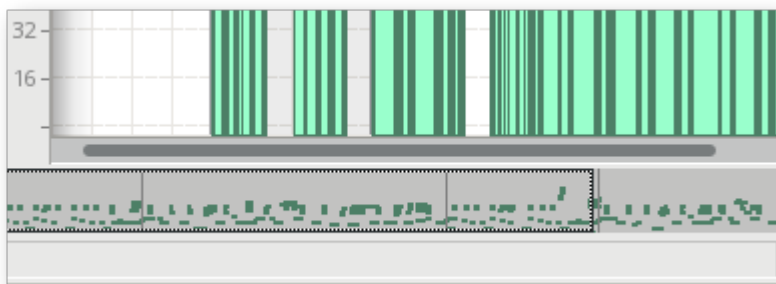
If you have no USB MIDI controller, you can still create MIDI data using Qtractor’s *MIDI Editor* (sometimes called a “matrix editor” or “piano roll” in other music programs). Data is edited on a grid through the usual GUI operations, such as selection, drag-and-drop, move, cut, copy, paste and deletion. The MIDI Editor can be invoked either for raw entry or to modify existing MIDI data that you’ve imported or played into Qtractor.

The MIDI Editor is very intuitive. The main, upper pane is the *Editor Grid*, which represents the notes being played. Vertically, the grid corresponds with **notes** on the chromatic keyboard (pitched as per the keyboard on the left of the window); horizontally, it corresponds with **beats** per bar. The lower pane is the *Value Grid*, which controls extra MIDI data, such as velocity, program changes, pitch bends and so on. The data represented here changes according to the *Value* menu setting in the upper menu bar.



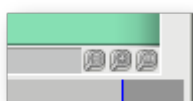
MIDI Editor window, showing notes in a familiar “piano roll” display (above) with their velocities (below)

You move around the MIDI Editor in the same way as the main work area - using the horizontal/vertical scroll bars at the bottom/right of the window, the mouse wheel/**Shift**+mouse wheel, or the navigation window (at the bottom of the MIDI Editor window itself).



The MIDI Editor’s navigation window, situated at the bottom of the MIDI Editor window itself

To zoom in/out, again, as per the main work area use the magnifying glass buttons at the bottom-right of both the Editor Grid and Value Grid (**Shift**-clicking or **Ctrl**-clicking if needed), or **Ctrl**+mouse wheel.



“Magnifying glass” zoom buttons, situated towards the bottom-right of the MIDI Editor window

All MIDI editing operations are available and processed in real-time. Several MIDI Editor instances may be active and open at any one time, provided each one refers to its own clip. All MIDI content may be saved as standard MIDI files (SMF) Format 0 (all tracks reduced to one track, preserving channel assignment data) or Format 1 (multi-track). The format for the SMF files may be set in **View / Options... / MIDI**.

To create new MIDI data:

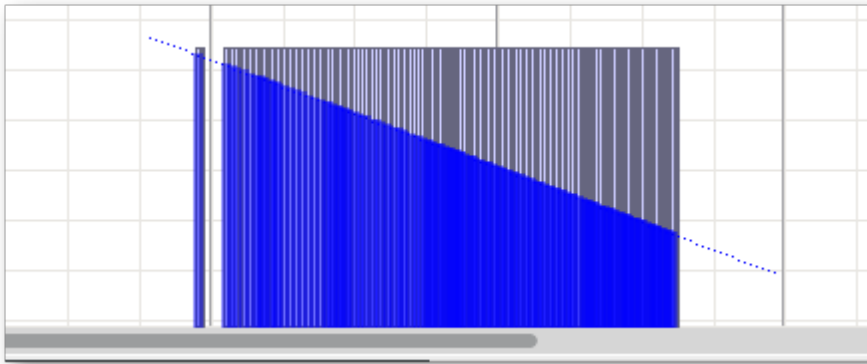
1. Create a destination [track](#), if you don't have one already
 - Right-click within the track and choose **Clip / New**. The MIDI Editor window will appear
 - Click the *Edit on* (pencil) icon in the menu bar and draw notes in the upper pane of the grid. You can also click the *Edit Draw* (pencil with wavy line) icon to draw multiple notes "free-hand" by dragging the mouse
 - In the bottom *Value Grid* pane you can adjust a variety of parameters. The default view shows *Note Velocity*, but this can be changed via the *Value type* and *Parameter type* combo boxes just below the pencil icons. Drag the bars up/down to increase/decrease their values, or click in a blank area (with one of the Edit modes enabled) to add new data

Below is a summary of the MIDI Editor's main tools:

- *Arrow* - select/move notes in the Editor Grid (or values in the Value Grid) and increase/decrease their length. To select all the notes of a particular pitch, left-click the desired key of the piano keyboard on the left hand side of the window and drag the cursor right, into the editor area
- *Edit on* (pencil) - draw notes or values, with their spacing determined by the *Snap/beat* setting (see below). You can also hold down Ctrl or Shift in this mode to put the Edit cursor into select mode temporarily - this is useful for quickly switching between inputting and selecting notes/information, as it avoids having manually to switch between the Edit and Arrow (described above) cursor modes
- *Edit Draw* (pencil with wavy line) - dynamically draw multiple notes/values with a click-and-drag of the mouse. This is, again, governed by the *Snap/beat* setting, but the faster you move the cursor the fewer the lines that will be drawn
- *Note Type* - "Note On" indicates that the notes you draw in the grid represent the length of the note. "Key Press" indicates that the notes represent a MIDI event without a defined duration, such as triggering an external MIDI sequence to begin playing
- *Value* - determines the information shown in the Value Grid. This may be note velocity, program changes, pitch bend or other extra MIDI data
- *Parameter type* - only available when certain *Values* are selected, such as *Controller*. Provides further information for the *Value* setting
- *Snap/beat* (musical note) - determines the resolution when drawing and editing notes/values. In a 4/4 session, for instance, a setting of *Beat* would cause a click in the Editor Grid to produce a crotchet (quarter note), *Beat/2* would produce a quaver (eighth note) and so on. Regardless of the default note length, a note can be lengthened/shortened by clicking and dragging one of its edges to increase/decrease its length. The resolution of such changes is also determined by the *Snap/beat* setting
- *Value ramp* - allows you to draw a continuous, linear slope of MIDI events in the Value Grid. This can be useful for operations such as creating crescendos/diminuendos or adjusting pitch bend

To use the value ramp: 1. Choose your desired *Snap/beat* setting 2. Enable *Edit on* and *Edit Draw*, then draw a series of events in the Value Grid (lower pane) 3. Hold down Shift or Ctrl (or switch to *Edit off*) and highlight these events 4. Disable *Edit Draw* (leaving *Edit on* enabled), then hold down the left mouse button and drag the dotted line over the events to set the angle of

the slope. You can not only shave the tops off the bars but extend their heights too. You should see something like the below as you drag



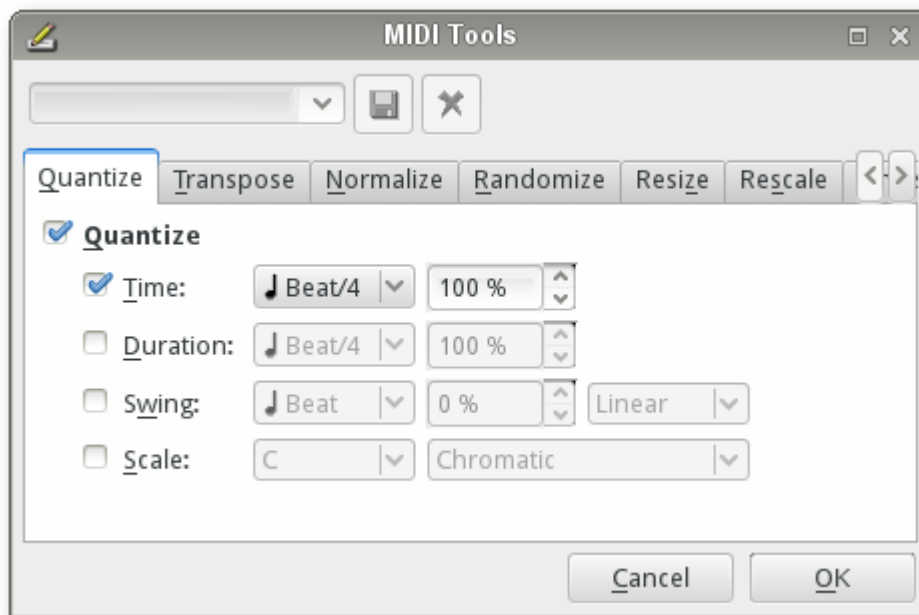
Adding a diminuendo using the value ramp

You can also use the tools detailed in [Additional MIDI Editor Features](#) with MIDI events such as these. For example, if you wish to make your diminuendo longer/shorter you can do so easily using the *Rescale* tool. With the MIDI events highlighted, right-click and choose **Tools / Rescale**, then adjust the *Time* percentage value accordingly.

4.10.5. Additional MIDI Editor Features

In addition to input and editing tools, the MIDI Editor features functions such as *Quantization*, *Transposition*, *Timeshifting* and more to make it easy to modify your composition. To access these features, open the **Tools** menu via the MIDI Editor toolbar, or by right-clicking within the window. If no notes are selected, the Tools will be unavailable; select a note or block of notes with the *Arrow* tool to make them available. Many of these tools are also very useful for adjusting MIDI event data in the Value Grid.

- *Quantize* - allows you to automatically structure notes in stricter uniformity with your time signature. This is especially helpful if you've played the notes in and were not perfectly on the beat. You can quantize notes such that the notes occur on the beat (or divisions thereof); such that the durations of notes are extended to be on the beat (or divisions thereof); such that notes play with some degree of "swing"; and even such that the notes played match a specific scale, from Minor to Major to the extremely obscure
- *Transpose* - allows you to transpose the pitch of, or shift the timing of, a note/notes, or reverse a sequence of notes
- *Normalize* - adjusts note velocity by either a percentage or an absolute value. A percentage value will adjust the velocity of the notes equally by the percent given in relation to the original velocity, while an absolute value adjusts velocity of each note to that value regardless of original velocity
- *Randomize* - provides random changes to the Note, Time, Duration, or velocity (Value). You may adjust how extreme the changes will be with percent values
- *Resize* - allows you to control the duration of notes to any number of beats (or divisions thereof), or the Velocity to any value
- *Rescale* - changes the selected notes by some percentage. You may alter the time that the notes are triggered, the duration for which they sound and the velocity at which they are played
- *Timeshift* - alters the timing of the selected notes on a curve, such that the acceleration from the beginning note to the ending note is either increased or decreased



The MIDI Editor's Tools menu

In order for Qtractor to know how to accelerate the timing of the notes, you must define a range of time for the acceleration to occur. Do this with the *Edit-head* and *Edit-tail* markers (the blue markers in the timeline above the MIDI Editor grid). Mark the out point (tail) of your range by right-clicking on the timeline, and then set the in point (head) of the time range by left-clicking somewhere to the left of the original marker. Once the range is defined, select the notes you wish to timeshift with the Arrow tool. Open the *Tools* menu and select *Timeshift*. Use the slider to cause either an acceleration, or a deceleration, in your selected notes over the course of the defined time range and then click **OK** to commit the change.

The **View** menu provides customization for how the MIDI Editor is laid out and how the notes are presented to you. The default layout is clean and pleasingly minimalist, but take a look at some of the other options to see what works best for you:

- *Menubar* - turns off the top menu bar; use **Control+m** to toggle it on or off
- *Statusbar* - turns off the status bar at the bottom of the window
- *Toolbars* - defines what information is visible in the window. Choose from *File* (file opening and saving icons), *Edit* (cut, paste, undo), *View* (preview modes, snapping, quantization), *Transport* (fast forward, rewind etc.), *Scale* (key signature and scale information) and *Thumb* (thumb view at the bottom of the window)
- *Windows* - toggles on or off event information, a panel which provides detailed information on each MIDI event in the MIDI Editor grid
- *Tool Tips* - toggles whether tool tips are visible
- *Note Duration* - defines whether the note's duration is reflected in the length of the velocity bars at the bottom of the grid
- *Note Color* - assigns differing colours to each note; if this is not active, all notes appear as one colour

- *Value Color* - assigns the colours of the corresponding notes to the velocity bars; helpful when you have many overlapping notes but want to adjust their velocities
- *Drum Mode* - displays MIDI Note On events as diamonds instead of rectangles
- *Zoom* - zoom in or out on the grid
- *Snap* - define what division of the beat your notes snap to, or turn snapping off entirely
- *Scale* - change the key signature or type of scale being used
- *Refresh* - causes the MIDI Editor to redraw in the event of latent images
- *Preview Notes* - toggle whether or not you hear the notes of the scale as you input or move notes on the grid
- *Follow Playhead* - define whether the MIDI Editor grid scrolls with the playhead when you are playing the track

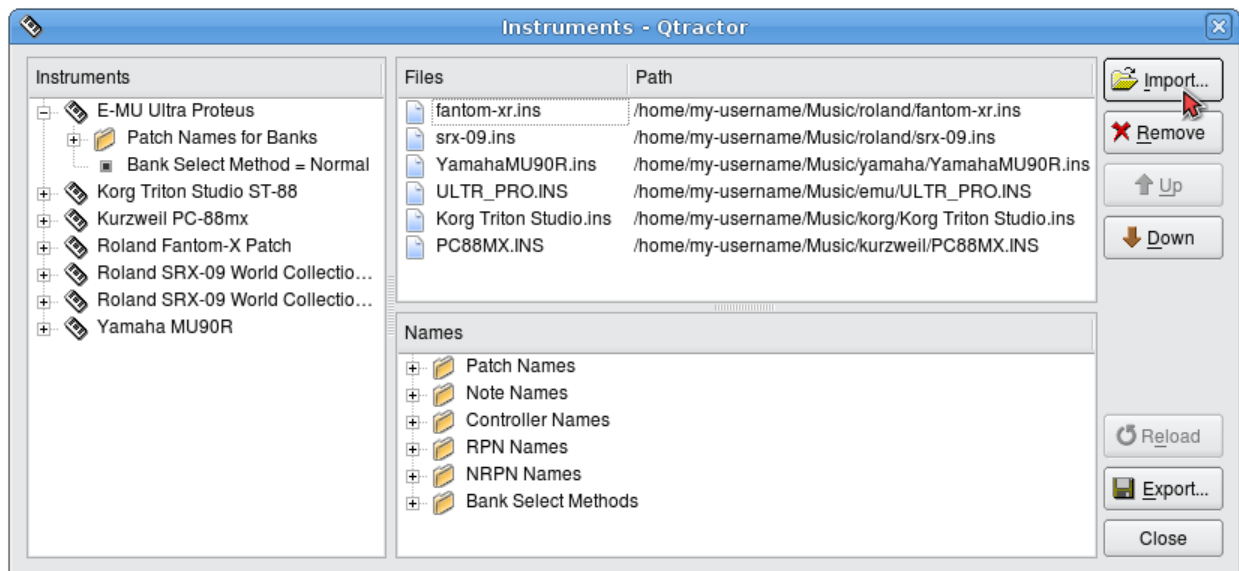
4.10.6. MIDI Instruments

There are many MIDI hardware tone generators available from a variety of manufacturers such as Yamaha, Roland and Korg. In Qtractor, information about the sound patches and sound banks of these tone generators is obtained from “instrument definition” (**.ins**) files. Qtractor supports instrument definition files used by the [Cakewalk](#)/Sonar MIDI sequencer software, offering a convenient MIDI bank-select/program-change mapping for existing MIDI instrument patch names and easier, intelligible selection of MIDI track channels. Files for many popular MIDI tone generators and synthesizers can be downloaded from the [Cakewalk site](#).

.ins files are imported via the **View / Instruments...** window. You can also import MIDI Name XML files (**.midnam**) in the same way. Once you’ve imported a file, select your desired patch via the **Bank** and **Program** options in the **MIDI / Instrument** area of the **Track** window. After clicking **OK**, the information will appear in the main view. Instrument patches can also be assigned via several other menus:

- **Track / Instrument** in the main toolbar
- **File / Track / Instrument** in the MIDI Editor
- Right-clicking a track in the left pane of the main view and selecting **Instrument**
- In the GUI of the plugin itself

When working with soundfonts (for example, via the [FluidSynth DSSI](#), or [Calf Fluidsynth LV2](#), plugins), you can also use the **View / Instruments...** window to import their patch names. Simply select the desired **.sf2/.sf3** file and its patch names will be imported. For DSSI plugins, changing the patch information in the **Track** window should update the relevant information in the plugin’s GUI automatically and vice-versa. However, this may not be the case for LV2 plugins (due to the LV2 spec, rather than any issue with Qtractor) - if there is a mismatch between the patch set in the plugin and the information in the **Track** window, upon re-loading the session the patch in the track may override that of the plugin. To avoid this behaviour, in the **Track** window set **MIDI / Instrument** to *(No instrument)* and **Bank** and **Program** both to *(None)*, then perform all patch selections directly in the plugin.



Instruments window, where MIDI instrument definition files, MIDI Name XML files and soundfont patch names can be imported, exported and moved

4.10.7. Soft Synths

Strictly speaking, software synths, or “soft synths”, are not part of Qtractor. However, it is common to use them within Qtractor. Acquiring free soft synths online is a simple matter of locating them via the project’s homepage, your distribution’s repositories, or sites such as sourceforge.net.

Soft synths for GNU/Linux come in a few varieties:

- Standalone synth applications that launch separately from Qtractor, but which plug into JACK so that they can be used as sound sources within Qtractor. One such example is [Qsynth](http://www.qsynth.org), written by the same programmer as QJackCtl and Qtractor
- DSSI Plugins, which are launched and controlled entirely from within Qtractor. For example, [Fluidsynth DSSI](http://www.fluidsynth.org)
- LV2 Plugins, which are launched and controlled entirely from within Qtractor. For example, [synthv1](http://www.synthv1.com)
- VST Plugins. The format from Steinberg does in fact support Linux technically, but there are few Linux-native VST synths



synthv1 - a soft synth available in both standalone and LV2 plugin forms

Once your synth of choice is installed, the method of launching it will depend on what form it is. If it's a standalone application, it can be launched like any other application on your system. If it's a plugin for Qtractor, launch Qtractor and create a new MIDI track, then choose your soft synth for the MIDI Instrument, as described [here](#).

4.10.8. Soft Synths as Plugins

Using soft synths as LV2 or DSSI plugins is probably the easiest model, as it does not require any additional routing through JACK. The synth is integrated into Qtractor's interface, the sound is directed to the appropriate track, and the fact that you are using a separate application is almost entirely abstracted away from you. For more information on using plugins, see the [Plugins](#) section.

Most soft synths have some level of control over the sounds they produce. To edit a synth's properties:

1. Choose **View / Windows** and open the **Mixer**
2. In the panel towards the top, locate your synth and make sure it's on (designated by a green lamp on the left of its name). If the lamp is black (synth turned off), click it so that it turns green
3. Double-click on the soft synth to open its settings window and change the desired parameters. The actual information presented will differ depending on the synth

Whether you use LV2, DSSI or VST plugins, the process for using a soft synth (or plugin effect) is the same. In addition, all parameters provided by the interface can be automated, enabling attributes like LFO or resonance to change over time during playback. For more information on this see the [Automation](#) section.

4.10.9. Stand-alone Soft Synths

The other kind of software synth you might use is a stand-alone synthesizer, such as [Qsynth](#). To use such a synth, launch QJackCtl and Qtractor, then launch the synth the same as you would any other application on your system.

Once you've launched your synth it will become a source for audio and a destination for MIDI signals in Qtractor. However, you must first verify that your settings are correct:

1. Open the **View / Windows / Connections** menu
2. The **Audio** tab contains all possible sources of audio. If you've launched QSynth, for example, it will be listed in the left column. Also listed will be Qtractor itself, since it too produces sound. In the right column are the audio destinations available. At the very least, you should see Qtractor's *Master In* and your computer speakers (labelled as *System > playback*). By default, the sound from Qtractor's *Master Out* is patched to *System > playback* so that you can hear everything that Qtractor's Mixer manages
3. This is where the workings of your own studio set-up should be considered. The audio from your synth can logically be patched into either Qtractor's *Master In*, so that its sound can be recorded into an audio track, or *System > playback*, if you have an external mixer or recording device that will be recording it. To patch the sound from the synth to its destination, click the left channel of the synth output (in the left column), then the left destination channel (in the right column), then the **Connect** button. Repeat this for the right channel if stereo is required

This takes care of routing the audio; now you must route the MIDI signals controlling the synth. Here again, there is no "right" way to route the flow of data - you must be somewhat familiar with the equipment and workflow you are implementing and decide for yourself how you want it all to work together. However, a general method of working is described below:

1. Open the **View / Windows / Connections** menu
2. The **MIDI** tab contains all possible sources of MIDI signals. At the very least, your stand-alone soft synth will be listed in the left column. If you have a USB MIDI controller, it too will be listed. Qtractor itself and a *MIDI Through* channel will also be listed. In the right column are the sound destinations available. At the very least, you should see Qtractor's *Master* and *MIDI Through*, together with any soft synths you are running
3. Connect the MIDI signal from Qtractor (in the left column) into your soft synth (in the right column) so that the data in your MIDI tracks will trigger the synth's sounds
4. If you are using an external USB MIDI keyboard controller, you will also likely want to route its signal into Qtractor's MIDI input so that you can input notes directly into your Qtractor MIDI tracks

4.11. Working with Audio

4.11.1. Audio Summary

There are two typical scenarios for dealing with audio in Qtractor and, indeed, for most digital audio workstations:

- Import existing audio files from your hard drive, a recording device, or a loop collection

- Record audio into Qtractor via your computer's built-in microphone or an external microphone

These methods of acquiring sound for your work can be, and very often are, combined to create a richer audio production.

4.11.2. Importing Audio Files

The easiest way to get sound into Qtractor is to import an audio file from your hard drive. Whether you are using sampled loops to construct a new musical piece, or importing a performance transferred from a recording device, importing audio in this way does not involve recording sound directly into Qtractor.

To bring audio into your project file, right-click in the **Files** pane and select **Add Files** or use **Ctrl+F**. Choose which file or files you wish to import from the file chooser window that appears. To place an audio file in a track, drag and drop it from the Files panel into the workspace. You can add it to an existing track, or drop it directly into an empty workspace area and a new track will be created automatically.

Note that when importing an mp3 file and adding it to the timeline, you may find that the waveform shows extraneous silence at the end. This is a known problem with mp3 files and it is especially prevalent with those encoded in VBR (variable bit rate) mode. To fix this, just delete the file from the timeline, then re-drag it back onto the timeline and the waveform should display correctly.

4.11.3. Recording Audio

Recording audio into Qtractor requires a microphone and at least one audio input channel. Many laptops and webcams have built-in microphones, so, in theory, you could use this as an input source, but for best quality, purchase an external microphone and use it as your audio capture device.

For simultaneous multi-track recording, rarely will the sound card that was bundled with your computer be sufficient. Almost all sound cards embedded on motherboards are set to mix the input signals down to a stereo mix. If you wish to record three separate musicians at the same time, each to a separate track in Qtractor, you will require a sound card with separate, dedicated inputs.

The way that Linux displays available sound inputs and outputs can be daunting at first, until you understand the logic behind it. A Linux system displays sound devices the same way it displays hard drives and available network interfaces: the first device (regardless of actual inputs or outputs available via that device) is labeled **hw0**, the next **hw1**, the next **hw2**, and so on. It is safe to assume that **hw0** would be the built-in sound card on the system; being embedded in the motherboard would certainly qualify it as being the first available sound device.

So, **hw0** represents, in almost every case, your built-in sound card. **hw1** might represent, for instance, a webcam that you keep plugged into your desktop, and **hw2** could represent, perhaps, a USB microphone or a USB interface that you've plugged in. You can usually determine which device is which by looking at the vendor name associated with the **hw** labels; for example, if you have two devices plugged into your system and one is labeled **Blue** and the other **H4**, then you would know from these terms that one is your Blue USB microphone and the other your Zoom Studio H4n.

With regard to outputs, these can sometimes be confusing due to the many possible ways you may wish to output your sound. Typically the stereo mix of your system sound is available as the first two output devices. If you have more than just two speakers and you wish to split your sound to each, then utilize the outputs labeled appropriately (Front, Center and so on). If the sound input and output labels confuse you, take a few minutes to learn them by playing sound on your computer and plugging speakers into each output on your computer. For the inputs, plug a microphone into your different inputs and see where they are received and how they are labeled. It won't take long before you understand the logic behind the labels, and you'll be able to use Qtractor all the more fluidly.

4.11.4. Recording from Line-In

There are many kinds of microphones, each intended for certain kinds of sounds and situations, but on a purely technical level, without the question of aesthetics and microphone design, there are only about three scenarios you will encounter:

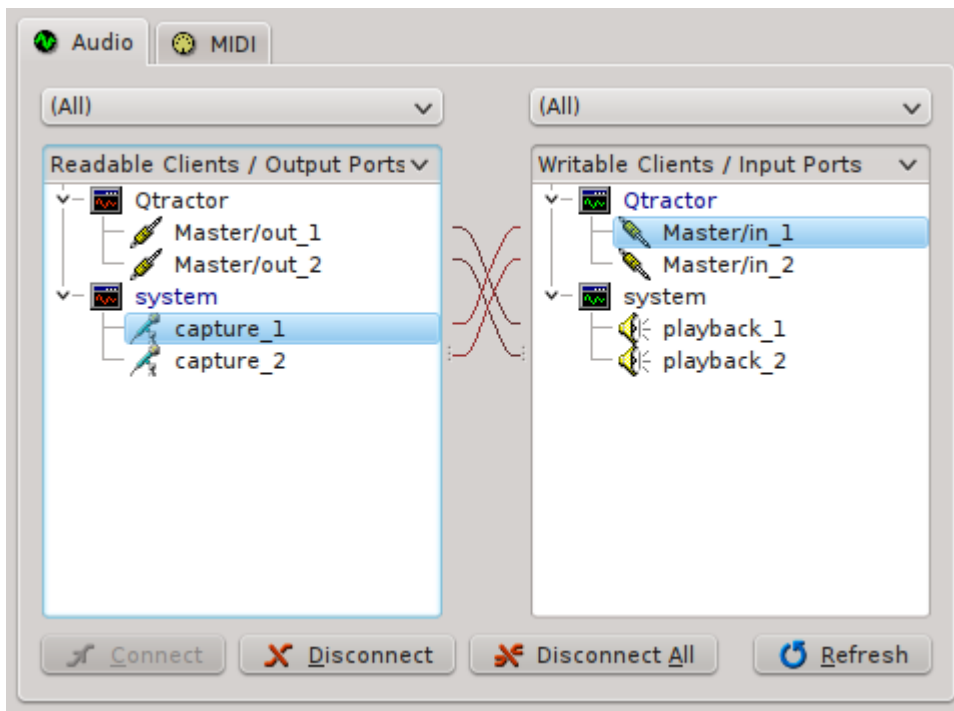
- Microphones with an 8th-inch (also called a mini) jack
- Microphones with a quarter-inch or XLR jack
- Microphones with USB connectors

The recording process is different, depending on your input type.

If your microphone has an 8th-inch jack, then you can plug it directly into the line-in of your computer. No external sound interface is required. Your built-in sound card is JACK's default input, so no changes are necessary. If your microphone can easily and cleanly adapt to the standard 8th-inch input with a cable or a simple plug adapter, then you can use this method of recording, as well.

To record from the line-in of your computer into Qtractor:

- Go to the **View** menu and select **Windows / Connections** to verify that the *Capture* devices on your System are routed to the *Master/In* of Qtractor and that the *Master/Out* channels are routed to the *Playback* channels of your System



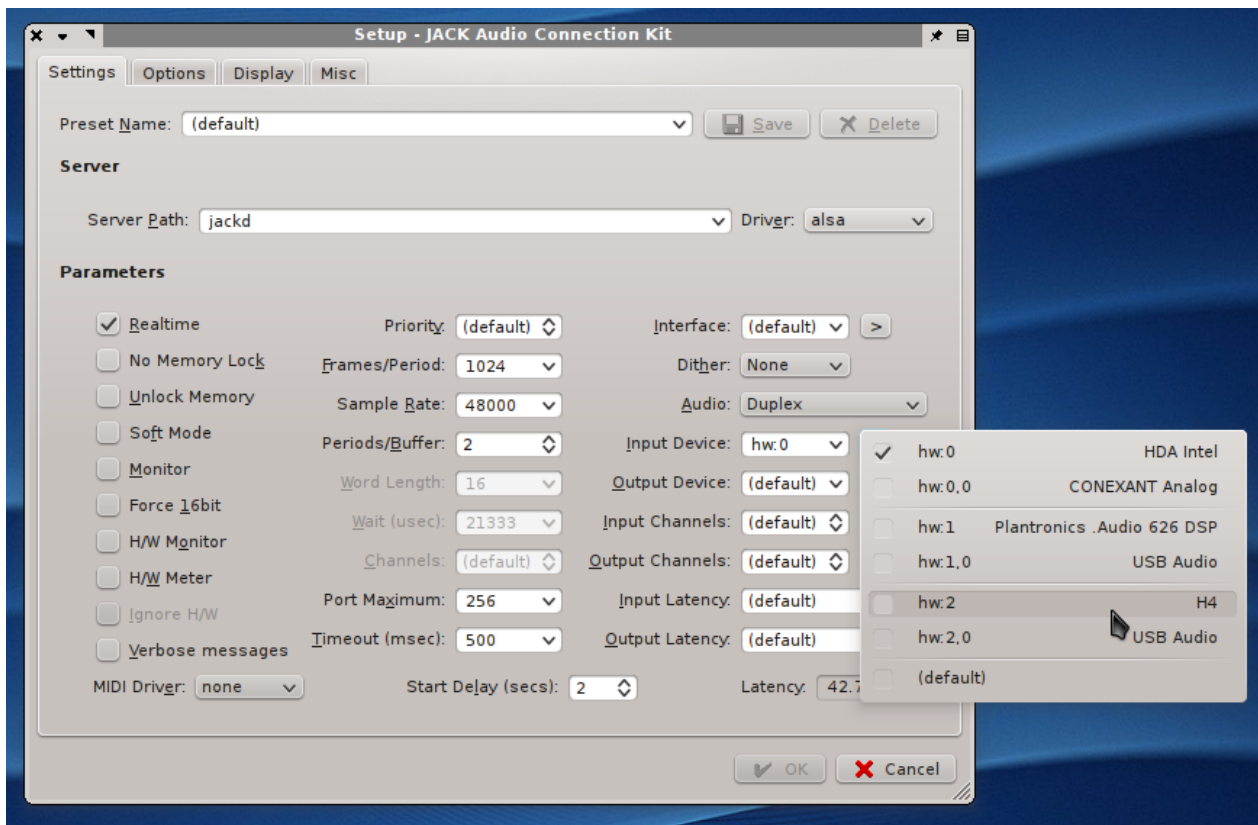
Connections window, set up to capture audio

- Go to the **Track** menu and choose **Add Track** or use **Ctrl+Shift+N**. In the **Track** dialogue box, give the track a name, set the *Type* to *Audio*, and set the *Input/Output* to *Master*. Click the **OK** button to proceed
- Arm the new track for recording by clicking the “R” button in the track listing on the left of the main Qtractor window. This sets the destination for the recorded sound
- Click the *Record* button in the top toolbar
- Click the *Play* button in the top toolbar

4.11.5. Recording from USB

A USB microphone plugs directly into the USB port of your computer. If you input sound through USB, then (obviously) you are utilizing a different interface than your computer’s built-in sound card. This must be set via QJackCtl for appropriate sound routing to occur:

- In QJackCtl, stop the sound server by clicking the Stop button
- If you have not already plugged in your USB audio interface or USB microphone, then do so. Make sure it’s on
- Click the **Setup** button. In the **Settings** tab, locate the *Input Device* setting and click the “>” button to see your choices

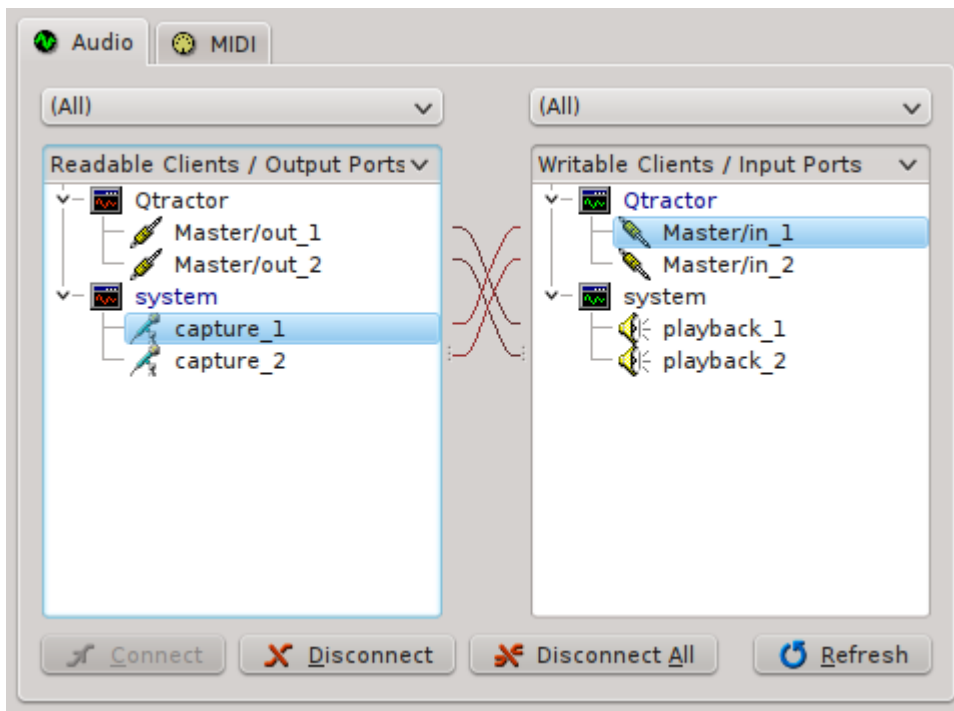


QJackCtl settings window showing Input Devices

- Click the **Start** button to activate

Now your USB interface, whether it is a single USB microphone or a 4-channel Audio-to-USB conversion box, is managing your input sources. Plug your XLR microphone into your USB interface, set the interface's input source as your XLR jack, and then create a new track in Qtractor and begin recording:

- Go to the **View** menu and select **Windows / Connections** to verify that the *Capture* devices on your System are routed to the *Master/In* of Qtractor and that the *Master/Out* channels are routed to the *Playback* channels of your System.



Connections window, set up to capture audio

- Go to the **Track** menu and choose **Add Track** or use **Ctrl+Shift+N**. In the **Track** dialogue box, give the track a name, set the *Type* to *Audio*, and set the *Input/Output* to *Master*. Click the **OK** button to proceed
- Arm the new track for recording by clicking the “R” button in the track listing on the left of the main Qtractor window. This sets the destination for the recorded sound
- Click the *Record* button in the top toolbar
- Click the *Play* button in the top toolbar

4.11.6. Recording Takes

Aside from one-take recordings, you can create multiple, continuous takes by using the Loop recording mode. To enable the Loop recording mode, go to **View / Options...** and, on the **General** tab, set *Loop Recording mode (takes)* to either ‘First’ or ‘Last’. This will control whether Qtractor chooses the first or the last take by default.

After you have enabled the Loop recording mode, select a loop range. To do so, position the Edit markers by left-clicking on the timeline where you want your loop range to start and right-clicking where you want it to end. Now select **Transport / Loop** (or press **Ctrl+Shift+L**). This will create a loop range out of your edit range and enable the loop mode as well. Just arm the track you want to record to, press *Record*, then press *Play*. Qtractor will start recording, but when it reaches the end of your loop range it will wrap around and start recording a new take of your loop range. You can record as many takes as you wish and decide later which one you like the most.

After having recorded some takes, select your new clip and go to **Clip / Takes / Select** to choose between the different takes. You can also press **Shift+T** to iterate through the recorded takes.

It is also possible to combine Punch in/out recording with looped takes recording. Just create a new Loop range as before, then create a Punch in/out range inside your loop range. Recording

now starts as soon as the playhead enters your Punch in/out range. Recording stops once the playhead leaves the range, but it will start again, recording another take, as soon as your loop starts over and reaches the punch in point again.



Punch in/out range (purple markers) inside a loop range (green markers)

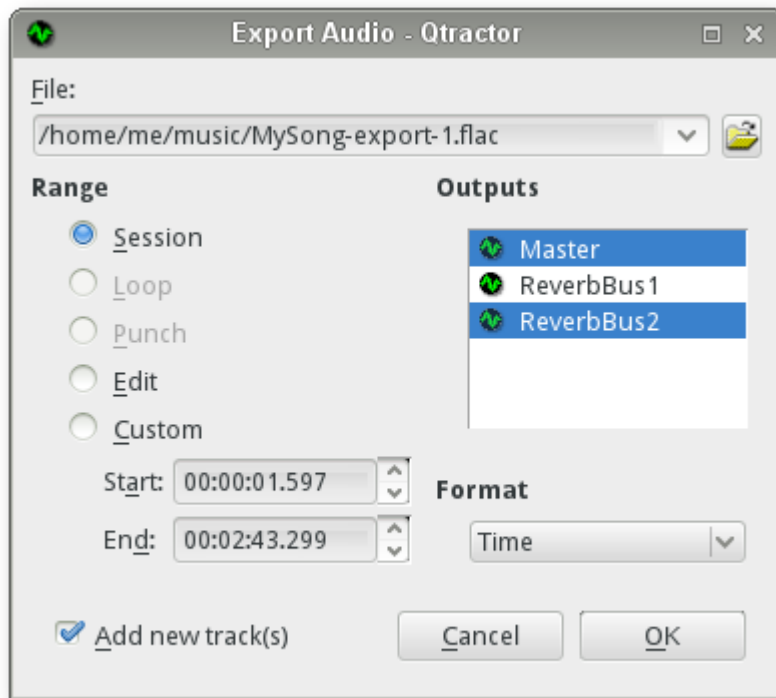
Takes are implemented in such a way that they are appended to each other. In other words, multiple recorded takes will result in a single clip that contains all takes appended to each other. By having activated the takes mode, Qtractor will show you only a section of this clip with the size of the selected loop range. If you go to **Clip / Take / Select** and choose 'None', Qtractor will 'unfold' your takes and show the recorded clip at its full length.

Once you know how takes are implemented, you can create them from any clip you like. For example, select a clip and place the edit markers within a fraction of your clip. If you have a clip that is four measures long, place the first edit marker at the beginning of the clip and the last at the beginning of the second measure. Now go to **Clip / Take / Range**. In the dialog box which appears, choose 'Edit'. The box on the right of this will show the resulting number of takes - in this case it will create four takes (one for each measure). Just click on the take you want to select. As you can see, takes can be created with a variety of ranges - just use the one that fits your needs best.

Finally, be aware that take state information is kept *only for the original recording track location*. If you change or cut any of the folded clip-takes all take state information will be lost (though such operations can be undone, as normal). If you move a clip to another area of the same track, then select a different take, the clip will "snap" back to its original position. If you move a clip to another track, take state information will be lost. Note that even if information has been lost in whatever way, you can still drag the clip's edges to access any hidden takes manually.

4.12. Audio / MIDI Export

All or part of the session may be exported to one audio or MIDI file. To export a file, select **Track / Export Tracks / Audio...** or **MIDI...**, or right-click in the tracks area of the main view to access the same menu. This will open either the **Export Audio** or **Export MIDI** window.



*Export Audio window, where the tracks in a session can be exported as a single audio file in the format previously specified by the user in **View / Options...***

The **Range** options enable you to specify how much of your session to export. You may export the whole **Session**; any **Loop**, **Punch** or **Edit** range you have previously denoted with its markers; or an arbitrary **Custom** period (the format of which can be expressed in **Frames**, **Time** or **BBT**).

The **Outputs** box allows you to choose which buses to export. **Ctrl**-click to select more than one. Note that when exporting multiple buses, because the buses' outputs are added together, the exported file can become clipped. To address this, you may choose to:

- Lower the level of the output buses, so that their combined output level does not exceed 0dBFS (monitoring this combined output via an external meter if necessary)
- Export to a float sample file format, by selecting one via the **Capture / Export** section of the **View / Options... / Audio** menu. This way, even if the exported file does become clipped, no data will be lost. You can then simply reduce the level of the file later using an editing tool, such as [Audacity](#)

Enabling the **Add new track(s)** option will automatically import a copy of your exported file into a new track below the currently selected one. This can be useful to “freeze” all or part of your session, mixing down several tracks to one new track in order to lighten CPU load or simply to tidy your workspace up.

MIDI export pertains to MIDI material only and results in the merging and concatenation of the selected MIDI tracks and clips into a single file of either SMF Format 0 (all tracks reduced to a single track) or SMF Format 1 (multi-track).

Audio export is implemented through the special JACK *freewheel* mode and is thus faster than real-time, resulting in a complete and exact mix-down of the selected audio material into a designated audio file of the selected format (wav, aiff, flac, au, ogg etc.). It is also possible to export the output of MIDI tracks directly as audio (with standard audio tracks also included in the exported file, if desired) using the same **Export Audio** window, provided that the MIDI tracks'

audio is generated by plugin instruments. For MIDI tracks which are used to send data to external instruments driven by JACK-MIDI or ALSA-MIDI (the output of these instruments then being routed back into Qtractor), it is necessary to “bounce” their output to audio tracks within Qtractor before export. The process of bouncing is covered in this [How To][How To - Sample MIDI Composition Workflow].

The user-preferred format for exported audio and MIDI files can be set in the Options window (**View / Options...**).

4.13. Snapshots, Templates and Archives

4.13.1. Snapshots

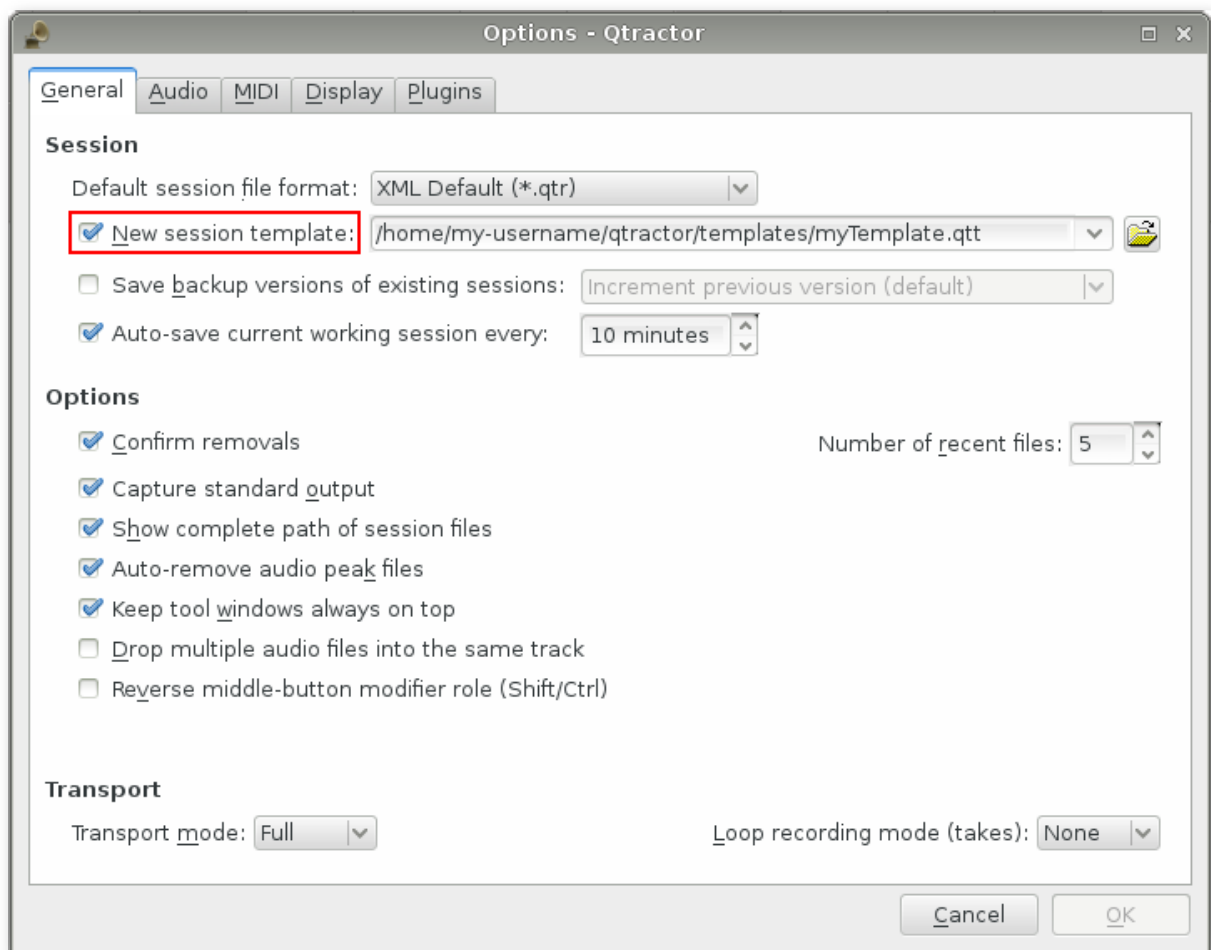
After the initial save of your project, you may find later that you want to save multiple “snapshots” of a session to compare it at different stages. To do this, click **File / Save as...** and Qtractor will add an incremental number to the end of your file name. Once saved, you can then switch freely between these snapshots by opening the relevant file.

Be aware that snapshots saved in the standard session file formats (**.qtr** or **.qts**, which are essentially the same) save only *references* to clips, rather than the clips themselves. Because of this, “destructive” edits (such as changing the notes of a MIDI clip) will be reflected in all snapshots. For example, if you have a MIDI clip in snapshot 1, then create snapshot 2 and make changes to the notes in that same clip, when you re-load snapshot 1, the changes you made in snapshot 2 will now be present in snapshot 1 as well. If you wish to avoid this and preserve all of your clips in a separate snapshot, use the **.qtz** archive format, which will safely bundle its own versions of all the clips (see also [4.13.3. Archives](#) below).

4.13.2. Templates

Templates are an extremely useful time-saving feature, which enable you to restore many settings automatically for use in a new project, thereby eliminating the need to re-configure tracks, buses, connections etc. each time. To create a template, once you have your session set up the way you want it, choose **File / Save As** and save the file in the **.qtt** format. This file can then be loaded in the same way as a standard session. Template files reproduce all the original session’s settings, without any MIDI or audio data, leaving you ready (or at least part of the way ready) to begin a new project. Exactly what you include in your template will depend on your needs, but you’ll begin to discover what these are as you use Qtractor. You can save any number of templates to cover whatever kind of projects you need.

If you find that you’re using the same template often, you may choose to have it auto-loaded for every new session. To enable this, choose **View / Options...** and, in the **General** tab, specify your template in **New session template**.



The Options window

4.13.3. Archives

Qtractor has a zipped archive format, **.qtz**, which collects together all the information needed to reproduce the session elsewhere, such as settings, connections, plugins (LV2 only) and all MIDI and audio files (minus any which have been disassociated with the session via the **Cleanup** option in the **Files** pane). This format can be used to share your Qtractor project with other people, or simply to tidy up a session which contains extraneous files.

Note that even if you save an archive with a different file name the original session's Properties will remain as they were, meaning that any clips subsequently recorded will still bear this old name in their file name. For example, if your original session has "Song_A" as the Name in its Properties and you save an archive of, then work on, the new session as "Song_B", all new clips will continue to be called **Song_A-trackName-1.mid** etc. If you want the new session name to be reflected in the file names of both new and existing clips, open the **File / Properties... / Session** menu and update the name in the **Name:** box. Qtractor will then update the session's midi and audio clip file names to match this name.

4.14. Other Qtractor Features

4.14.1. Metronome

When recording audio or MIDI data in real-time, a metronome can be very useful for keeping your performance in time. Qtractor has both audio and MIDI metronomes. The audio metronome generates its sound within Qtractor; the MIDI variant requires an external sound source, such as a standalone soft synth. You can even use both metronomes at the same time if desired.

To set up a metronome, open the **View / Options...** menu.

- **Audio metronome** - select the **Audio** tab. In **Metronome**, tick *Enable audio metronome* and point Qtractor to the files you wish to use for *Bar* and *Beat*. You can also adjust the *Gain* if necessary, plus create *Dedicated audio metronome outputs* (useful if you have a particular monitoring set-up in mind)
- **MIDI metronome** - select the **MIDI** tab. In **Metronome**, tick *Enable MIDI metronome*, select a *Channel* and specify the *Note (bar)* and *Note (beat)* you wish to use. You can also adjust the *Velocity* and *Duration* of these notes. The MIDI metronome triggers sounds via the *MIDI Master Out* bus by default, but if you'd prefer to use a separate bus you can enable *Dedicated MIDI metronome output* - this will then appear as *Metronome* in the MIDI tab of the [Connections](#) window. Once you've decided which bus to use, connect it up to your destination via the Connections window

If you're working with noticeable latency in your system, the *Offset (latency)* option, available for both audio and MIDI metronomes, allows you to compensate for this.

Percussion sounds are often employed for a metronome, though what you use is entirely your choice. For the audio metronome, if you have no suitable files to hand you can search the internet or use the ones from [this thread](#) on rncbc.org. For the MIDI metronome, if you want percussion sounds and you're using a General MIDI-compliant bank, set *Channel* to **10**, which is the channel for percussion.

In order to hear either metronome while the transport is rolling you must turn it on. To do so, click the blue triangle icon in the top menu bar, or select **Transport / Metronome**.

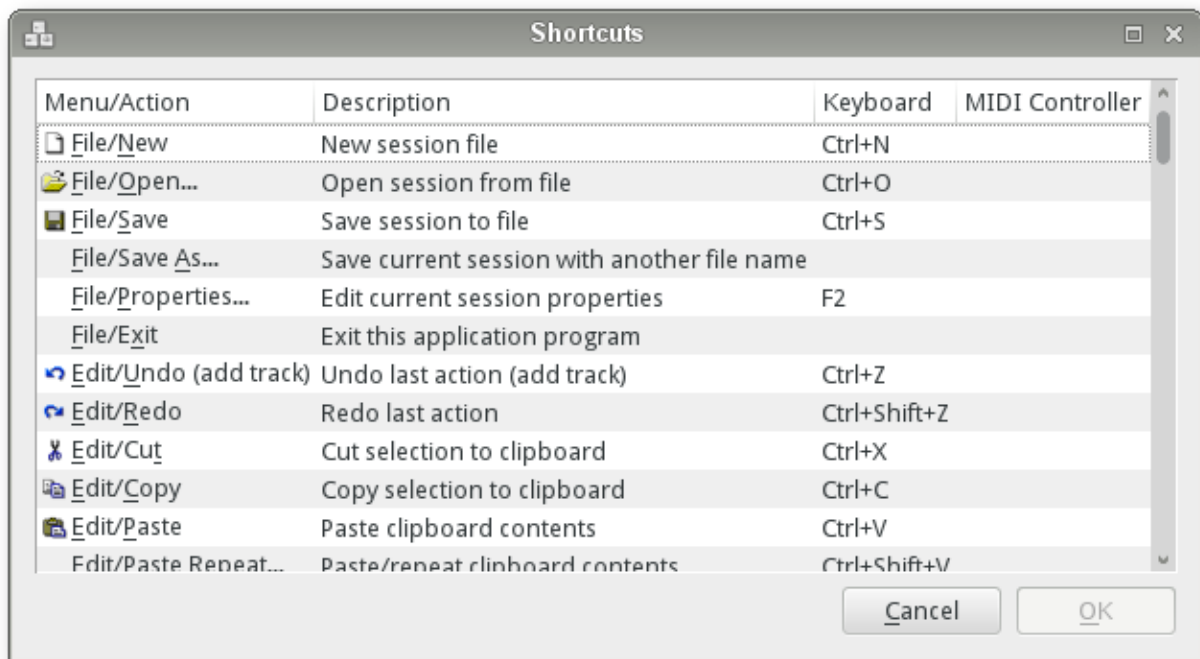


The Metronome icon

4.14.2. Keyboard Shortcuts Editor

Keyboard shortcuts are useful for the power user, in that they provide a quick mechanism for performing often used commands quickly, without the use of your mouse. Shortcuts can also be assigned for MIDI Controllers.

Keyboard shortcuts may be customized to your preference by using the shortcuts editor. This editor may be found in the Help menu (**Help / Shortcuts...**). Note that there are two separate shortcuts windows - a general one, accessed from the main window and a MIDI Editor-specific one, accessed from the MIDI Editor window.



*Keyboard Shortcuts Editor window, available from the **Help / Shortcuts...** menu item*

It is very straightforward in its use. Simply find the item you want to create a shortcut for, left-click in the shortcut cell, and choose 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. The “X” icon on the right of the window can be used to undo any changes or remove a default shortcut.

Take care when using shifted combinations, as if that combination produces a non-alphanumeric character it may not function as a shortcut. For example, **Shift+1** will likely end up as a symbol such as “!” in the menu and, upon attempting to use this combination as a shortcut, it will not work. **Alt** and **Ctrl**-modified shortcuts should, however, be fine.

Qtractor will display a warning message if you try to duplicate a shortcut in the same menu, but not if you use the same shortcut in both the main and MIDI Editor menus. Instead, a warning message will appear in the Messages window. This warning is driven by the Qt framework and is, unfortunately, outside of Qtractor’s control.

Manual - 5 Qtractor Menus

[Manual - Table of Contents](#)

5. Qtractor Menus

5.1. Main Workspace

File

- **New** – create a new session project
- **Open...** – open a previously created session project
- **Open Recent** – access a list of several of your last used session projects
- **Save** – save your current session project
- **Save As...** – save your current session project with naming conventions
- **Properties...** – open the session properties dialog
- **Exit** – exit the program

Edit

- **Undo** – undo the last action
- **Redo** – redo the last action
- **Cut** – delete and copy an item to the clipboard
- **Copy** – copy an item to the clipboard
- **Paste** – paste an item from the clipboard
- **Paste Repeat...** – paste an item from the clipboard multiple times, with a user-definable period in between each
- **Delete** – delete the selected item
- **Select Mode** – select Clip, Range or Rectangle edit modes, or Automation
- **Select** – select All, None, Invert, Track, Track Range or Range
- **Insert** – insert a range or track range
- **Remove** – remove a range or track range
- **Split** – after selecting a Range or Rectangle, split according to this selection
- **Track** – add, remove and adjust the properties of tracks

- **Clip** – create a new clip, edit an existing clip, or import/export a clip

Track

- **Add Track...** – add either a new audio or MIDI track
- **Remove Track** – remove a track
- **Duplicate Track** – create a copy of the currently selected track, directly beneath it
- **Track Properties...** – open the track properties dialog
- **Inputs** – show current track inputs
- **Outputs** – show current track outputs
- **Instrument** – select the instrument name of a MIDI track, from those defined in the View / Instruments menu
- **State** – set current track state: Record, Mute, Solo, Monitor
- **Navigate** – navigate through tracks: First, Previous, Next, Last, None
- **Move** – change position of current track: Top, Up, Down, Bottom
- **Height** – adjust the vertical height of a track
- **Auto-Monitor** – toggle current track auto-monitoring
- **Auto-Deactivate** – deactivate plugins not producing sound
- **Import Tracks** – import audio or MIDI file(s) as new track(s)
- **Export Tracks** – export audio or MIDI track(s)
- **Automation** – automate the parameters of a track or its plugins

Clip

- **New...** – add a new clip to the currently selected track, bounded by the Edit-head and Edit-tail markers (blue vertical lines)
- **Edit...** – edit the currently selected clip
- **Take** – select from multiple takes recorded, or set a range to create a new one
- **Unlink** – if a MIDI clip is copied within the same track, any subsequent changes will be reflected in both the original and copied clips. This unlinks the clips, deactivating this behaviour
- **Record** – set a clip to be overdubbed on the next recording operation
- **Split** – split the clip at the position of the play-head (red vertical line)
- **Merge...** – merge two or more currently selected clips into one

- **Normalize** – normalize the level of an audio clip to 0dB
- **Tempo Adjust...** – adjust the clip's tempo and time signature
- **Cross Fade...** – cross fade two overlapping clips
- **Range Set** – set the edit boundaries (blue vertical lines) to the start/end of the currently selected clip
- **Loop Set** – set a loop between the start/end of the currently selected clip
- **Import...** – import audio or MIDI file(s) as new track(s)
- **Export...** – export audio or MIDI track(s)
- **Tools** – access a variety of tools to manipulate a MIDI clip

View

- **Menubar** – toggle display of the menu bar. Restore with **Ctrl+M**
- **Statusbar** – toggle display of the status bar
- **Toolbars** – toggle display of various toolbars
- **Windows** – choose which windows are displayed: File System, Files, Messages, Connections, Mixer
- **Tool Tips** – toggle display of tool tips (despite the name, these are not tool tips for the GUI icons at the top of the screen, but for the information seen when hovering over a clip)
- **Zoom** – zoom the main view: In/Out/Reset for Horizontal axis/Vertical axis/All
- **Snap** – change the Snap/beat setting
- **Refresh** – refresh the view contents
- **Instruments...** – open the instruments dialog, where you can import/export patch names etc.
- **Controllers...** – manage MIDI controllers
- **Buses...** – open the buses editor dialog
- **Tempo Maps / Markers...** – manage tempo and time signature changes
- **Options...** – open the program options dialog (see below)

Transport

- **Backward** – move playhead backward to the beginning of a range or session
- **Rewind** – move playhead backward at greater than usual speed
- **Fast Forward** – move playhead forward at greater than usual speed

- **Forward** – move playhead forward to the next edit marker or end of the session
- **Loop** – repeatedly play the range marked as a loop
- **Loop Set** – mark selected range as a loop
- **Stop** – stop the transport when rolling
- **Play** – move playhead forward at normal speed to play all un-muted tracks
- **Record** – prepare to record audio or MIDI on record-enabled tracks
- **Punch** – prepare to record the selected punch range
- **Punch Set** – mark the selected range for punch recording
- **Metronome** – turn the metronome’s audio or MIDI output on or off
- **Follow Playhead** – enable view scrolling so that the playhead is always visible
- **Auto Backward** – have the playhead return to the beginning of a range or session automatically when it stops
- **Continue Past End** – have the playhead continue to move forward even after it reaches the end of the session
- **Mode** – set the JACK Transport mode: None, Slave, Master, Full
- **Panic** – stop any MIDI or audio that may be playing, either via the Tracks pane or the Files pane

Help

- **Shortcuts...** – list the keyboard key combination equivalents for various menu items, commands and MIDI Controllers. The content of this menu differs from that of the MIDI Editor’s shortcuts menu
- **About...** – view information about Qtractor
- **About Qt...** – view information about Qt, the graphics toolkit used for Qtractor’s user interface

5.2. MIDI Editor

File

- **Save** – save the current clip
- **Save As...** – save the current clip with naming conventions
- **Unlink** – if a MIDI clip is copied within the same track, any subsequent changes will be reflected in both the original and copied clips. This unlinks the clips, deactivating this behavior
- **Record** – set a clip to be overdubbed on the next recording operation

- **Properties...** – open the clip properties dialog
- **Range Set** – set the edit boundaries (blue vertical lines) to the start/end of the currently selected clip
- **Loop Set** – set a loop between the start/end of the currently selected clip
- **Track** – set the inputs, outputs and properties of a track
- **Close** – close the current clip

Edit

- **Undo** – undo the last action
- **Redo** – redo the last action
- **Cut** – delete and copy an item to the clipboard
- **Copy** – copy an item to the clipboard
- **Paste** – paste an item from the clipboard
- **Paste Repeat...** – paste an item from the clipboard multiple times, with a user-definable period in between each
- **Delete** – delete the selected item
- **Select Mode** – select On, Off or Draw edit modes
- **Select** – select All, None, Invert or Range
- **Insert** – insert a range
- **Remove** – remove a range
- **Tools** – access a variety of tools to manipulate the clip

View

- **Menubar** – toggle display of the menu bar. Restore with **Ctrl+M**
- **Statusbar** – toggle display of the status bar
- **Toolbars** – toggle various toolbars on and off
- **Windows** – toggle display of the Events window
- **Tool Tips** – toggle display of tool tips (despite the name, these are not tool tips for the GUI icons at the top of the window, but for the information seen when hovering over a note/MIDI controller data)
- **Note Duration** – visualise note durations in the bottom pane
- **Note Color** – set notes/values to be displayed in different colors according to pitch

- **Note Type** – display either Note On or Key Press events
- **Value Color** – set notes/values to be displayed in different colors according to velocity value
- **Value Type** – determine the information to be displayed in the bottom pane
- **Drum Mode** – displays MIDI Note On events as diamonds instead of rectangles
- **Zoom** – zoom the view: In/Out/Reset for Horizontal axis/Vertical axis/All
- **Snap** – change the Snap/beat setting
- **Scale** – lock the addition/editing of notes to a particular scale, from the common to the esoteric
- **Refresh** – refresh the view contents
- **Preview Notes** – toggle whether notes are audible when inputting/editing in the matrix
- **Follow Playhead** – enable view scrolling so that the playhead is always visible

Tools

At least one note must be selected for this menu to be active.

- **Quantize** – quantize the timing or scale of notes
- **Transpose** – transpose the pitch of, or shift the timing of, a note/notes; reverse a sequence of notes
- **Normalize** – normalize the velocity value of notes using a percentage amount or absolute number
- **Randomize** – randomize note pitches, times, durations or velocity values by percentage amounts
- **Resize** – resize note durations or velocity values
- **Rescale** – rescale note times, durations or velocity values
- **Timeshift** – alter the timing of notes on a curve, such that the acceleration from the beginning note to the ending note is either increased or decreased

Transport

See Main Workspace above.

Help

- **Shortcuts...** – list the keyboard key combination equivalents of menu items and other commands. The content of this menu differs from that of the main workspace's shortcuts menu
- **About...** – view information about Qtractor

- **About Qt...** – view information about Qt, the graphics toolkit used for Qtractor’s user interface

5.3. View / Options

- **General** – configure global options, such as UI behaviour, session templates, back-ups and transport mode
- **Audio** – set capture/export format, playback options and metronome
- **MIDI** – set capture/export format, playback options, control options and metronome
- **Display** – configure the look-and-feel of the UI, including colours and fonts, plus enable/disable logging
- **Plugins** – set plugin paths, toggle plugin editor GUIs, toggle dedicated audio outputs for tracks and enable/disable various experimental features

Manual - 6 Appendixes

[Manual - Table of Contents](#)

6. Appendixes

6.1. References

Qtractor - An Audio/MIDI multi-track sequencer

<http://qtractor.sourceforge.net/>

<http://sourceforge.net/projects/qtractor/>

Qt 4 - C++ class library and tools for cross-platform development and internationalization

<http://qt-project.org/>

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/>

libvorbis - 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/>

LADSPA - Linux Audio Developer's Simple Plugin API

<http://www.ladspa.org/>

QjackCtl - JACK Audio Connection Kit - Qt GUI Interface

<http://qjackctl.sourceforge.net/>

<http://sourceforge.net/projects/qjackctl/>

Cakewalk - Powerful and easy to use products for music creation and recording

<http://www.cakewalk.com/>

<ftp://ftp.cakewalk.com/pub/InstrumentDefinitions/>

SoundTouch - Sound Processing Library

<http://www.surina.net/soundtouch/>

rncbc.org - The personal web presence of Rui Nuno Capela, creator and developer of Qtractor and other applications or application GUIs such as QjackCtl (a GUI for jackd), Qsynth (a GUI for fluidsynth), Qsampler, etc.

<http://www.rncbc.org/>

Manual Draft - Loop Mode Recording

!Warning - Draft of feature under development!

The features described below are not available in currently released versions of qtractor yet. If you want to use them, you need to build your own version of qtractor from the latest svn-revision. The features may also undergo changes as well as the documentation below.

Loop Recording

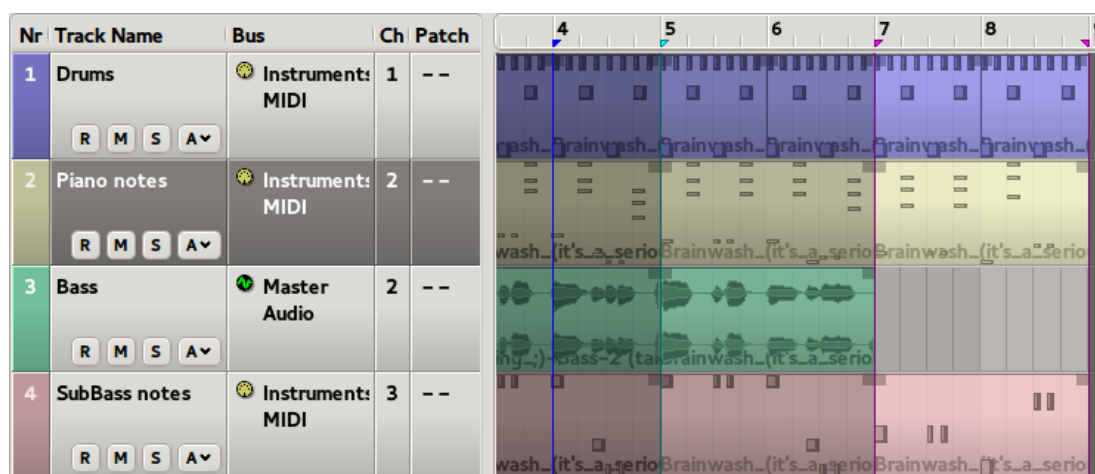
Aside of one take recordings, you can create multiple takes continuously by using the loop recording mode. To enable the loop recording mode, go to View > Options and on the 'General' tab set 'Loop Recording mode(takes)' to either 'first' or 'last'. This will control whether qtractor chooses the first or the last take by default.

After you have enabled the loop recording mode select a loop range. To do so, position the edit markers by left-clicking onto the timeline where you want your loop range to start and right-clicking at the end. Now select Transport > Loop (or press Ctrl+Shift+L). This will create a loop range out of your edit range and enables the loop mode as well. Just arm the track you want to record to, press record and play. Qtractor will start recording but when it reaches the end of your loop range, it will wrap around and start recording a new take of your loop range. You can record as many takes as you wish and decide later which one you like the most.

After having recorded some takes, select your new clip and go to Clip > Takes > Select to choose between the different takes. You could also press Shift+T to iterate over the recorded takes.

Advanced Loop Recording

It is also possible, to combine punch in/out recording with looped takes recording. Just create a new Loop range as before. Now, create a punch in/out range inside your loop range. Recording now starts as soon as you enter your punch in/out range. If you leave the range, recording stops, but will start recording another take as soon as your loop starts over and reaches the punch in point again.



punch in/out range inside a loop range

The images above shows a punch in/out range(purple markers) inside a loop range(green markers).

Unfold your Takes

Takes are implemented in a way, that they are appended to each other. In other words, multiple recorded takes will result in a single clip that contains all takes appended to each other. By having activated the takes mode, qtractor will show you only a section of this clip with the size of the selected loop range. If you go to Clip > Take > Select and choose 'None', qtractor will 'unfold' your takes and shows the recorded clip in its full length.

Create takes from preexisting material

Once you know how takes are implemented, you can create takes from every clip you like. For example, select a clip and place the edit markers within a fraction of your clip. If you have a clip that is four measures long, place the first edit marker at the beginning of the clip and the last at the beginning of the second measure. Now go to Clip > Take > Range. In the upcoming dialog, choose 'Edit'. The box on the right of this dialog will show the resulting number of takes. In this case it will create four takes(one for each measure). Just click on the take you want to select. As you can see, takes could be created by a variety of ranges. Just use the one that fits your needs best.

How To - Contents

How To's

- [How To - 1 Compile Qtractor](#)
- [How To - 2 Create Individual MIDI and Audio Buses-Ports](#)
- [How To - 3 Set the number of channels an audio track records](#)
- [How To - 4 Sample MIDI Composition Workflow](#)
- [How To - 5 Sidechain Workflow with Carla Plugin](#)
- [How To - 6 Alternative Sidechain Workflow with Aux Sends](#)
- [How To - 7 Equal Latency for Tracks and Buses](#)
- [How To - 8 Controlling a Plugin's Parameter from a MIDI Clip](#)
- [How To - 9 Distributing Plugins' Load to multiple CPU Cores](#)
- [How To - 10 Get SOLO functionality on Buses](#)
- [How To - 11 Prevent Color Picker Freezing Qtractor](#)
- [How To - 12 Automate Buses Experimentally with MIDI filters](#)
- [How To - 13 Make Successful Audio Connections](#)
- [How To - 14 Automate Buses and Tracks by Layer with Insert Controller](#)
- [How To - 15 Use Multiple SoftSynths on a Track](#)
- [How To - 16 Obtain a Multi Ghost Reference](#)
- [How To - 17 Get Individual Drum Instruments with a MIDI Track](#)
- [How To - 18 Take Advantage of Qtractor's Hidden Tricks](#)

How To - 1 Compile Qtractor

[How To - Contents](#)

- Introduction
 - What is compile?
 - Advantages
 - Dependencies
 - Git
- Compile
 - Install binary libraries
 - Download source code from official Git
 - Build the binary
 - Install

Introduction

What is compile?

Compiling is the process of translating human-readable source code into binary code readable by your hardware.

Go from having text files to an executable.

It is the way to install a program directly from its source code.

An installer file (.deb for example) works differently. Install previously compiled binary files. Distribute and configure executables on your operating system.

Advantages

Optimized executable: By compiling we ensure that the machine code is generated for our specific hardware, and not generic hardware. This should theoretically improve the efficiency of the program, although in practice I have not noticed any differences.

Have the latest version: If an installer with the latest version is not found in the repositories of our distribution, compiling allows us to install it.

Test new features: This way we can experience functionalities in development that are not yet in the official version.

Personalize: By having the source code, we can customize Qtractor before compiling: Our own icons, menus, functions, etc. The possibilities are endless.

Dependencies

There are two types of dependencies:

- **Libraries in source code:** They are resolved with GIT.
- **Libraries in binary code:** They are resolved with repositories of your distribution.

Git

The official Qtractor repository is located on sf.net, and it works with git.

Git is a version management system. It allows us to do things like:

- Download a project.
- Switch between branches of the same project.
- Resolve source code dependencies
- etc.

Compile

Install binary libraries

We will use as an example the Debian command “sudo apt install”. If your Linux is not based on Debian, you will have to adapt it to the order of your distribution.

The name of the libraries may vary between distributions.

If you work with kxstudio repositories, temporarily disable them before executing the command. The version of “lv2-dev” in this repository may create conflicts.

Update repositories

From the terminal:

```
sudo apt update
```

Install binary dependencies

If you are compiling Qtractor under a Debian or Ubuntu-based distro you can install all the required build dependencies with one command by running:

```
sudo apt-get build-dep qtractor
```

If it doesn't work for you, try:

```
sudo apt install qt6-base-dev qt6-base-dev-tools qt6-tools-dev qt6-tools-dev-  
tools qt6-l10n-tools qt6-svg-dev libqt6svg6-dev libjack-jackd2-dev libasound2-  
dev libsndfile-dev libvorbis-dev libmad0-dev libz-dev libsamplerate-dev  
librubberband-dev libfftw3-dev libaubio-dev ladspa-sdk dssi-dev liblo-dev lv2-  
dev liblilv-dev libsratom-dev libsord-dev libserd-dev libgtk2.0-dev  
libgdkmm-2.4-dev
```

Download source code from official Git

```
git clone --recursive https://git.code.sf.net/p/qtractor/code qtractor-git
```

The “--recursive” subcommand allows you to resolve source code dependencies. In this case for the CLAP and VST3 plugins. Git is thus responsible for attaching a copy from the repositories of these external projects.

Enter in the generated folder:

```
cd qtractor-git
```

Build the binary

Collect all the files and check if everything is in order

```
cmake -B build
```

Build

```
cmake --build build
```

If everything went well, you now have a functional executable. However it will not communicate well with the operating system. For example, it will show the interface in English, even if your Linux is configured in another language.

The executable can be found in the folder:

```
qtractor-git/build/src/qtractor
```

If you execute the previous line in the console you will see that qtractor opens and works correctly.

Install

Maybe you want to install it to enjoy Qtractor in your language.

```
sudo cmake --install build
```

By default it will be installed in:

```
/usr/local
```

Install to custom destination

If you want to install Qtractor in another destination you must modify the cmake:

```
cmake -DCMAKE_INSTALL_PREFIX=YourDestinationPath -B build
```

```
cmake --build build
```

```
sudo cmake --install build
```

How To - 2 Create Individual MIDI and Audio Buses-Ports

[How To - Contents](#)

You can create individual input, output, or duplex **MIDI/audio buses** (with corresponding ports) that can be assigned, after they are created, to individual tracks in the **Track Properties menu** for each track.

To do so, from the **Main** menu go to **View > Buses**, which opens up the Buses menu.

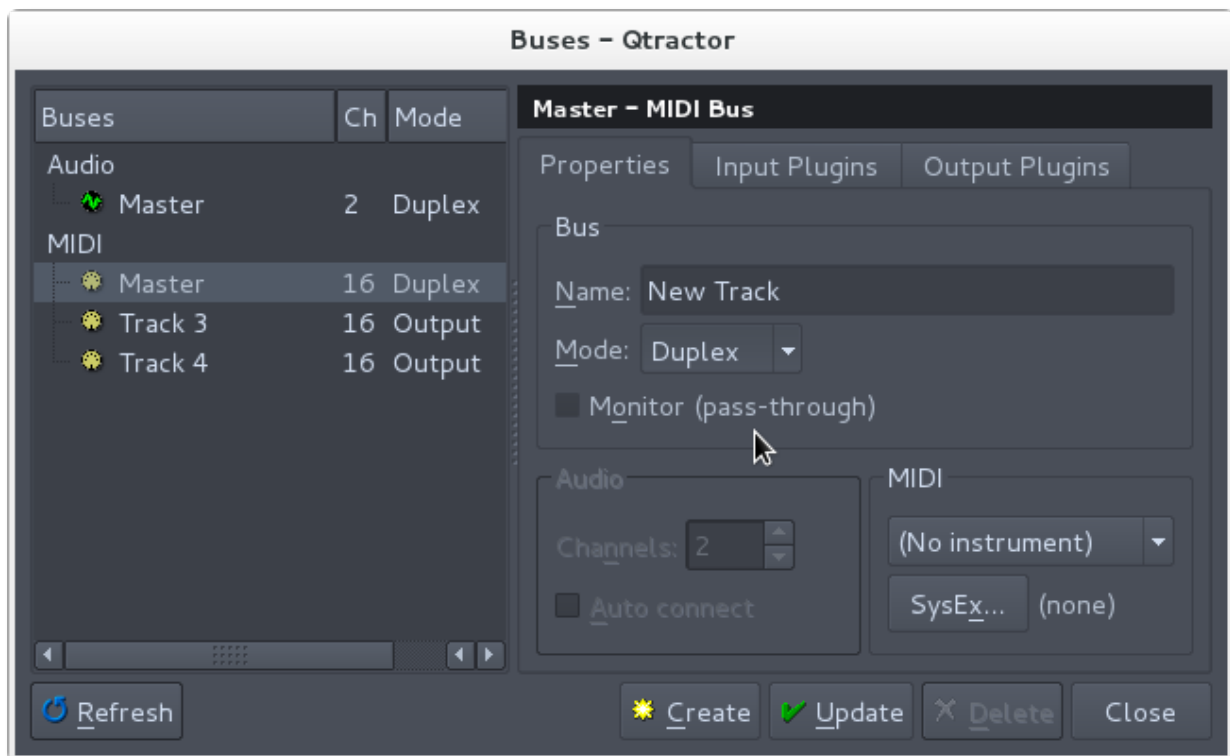


Illustration 1.1: Buses Menu showing audio & MIDI buses, Properties tab, Plugins tabs, Create and Update buttons

Select one of the buses of the type of bus that you want to create (MIDI or Audio) and it's properties will be shown. Once you change the **name**, the **Update** and **Create** buttons will be available. You can update the currently selected bus or create a new one with the name you provided. You can further change other options that are visible in the menu.

You can now assign them as you like in the **Track Properties** menu for each track.

Using individual MIDI buses for each MIDI track enables you to assign the full range of the 16 MIDI channels through one bus allowing you to avoid having to select a different channel for each softsynth or external MIDI device as you would have to do, if you are just using the Master MIDI bus. This also allows you to use one MIDI bus, connected to one external MIDI port, for different tracks for each instrument on a multi-instrument synthesizer or sound module. The same would be true for a multi-voice/instrument softsynth.

For further details, see manual section [4.1.1 Routing General Concepts and Information](#).

How To - 3 Set the number of channels an audio track records

[How To - Contents](#)

The number of channels that an audio track will record is determined by the channel count of its input bus. If you use the **Master In** bus (with its default setting of 2 channels) as the input bus for a track, the track will record two channels of audio data. If you're recording from a single channel source, such as a typical microphone or electric guitar pre-amp, you should [create an input bus][How To - Create Individual MIDI and Audio Buses-Ports] with a single channel, making sure it's connected to the external source you want to record, and then select it as the input bus in the track properties. This will result in a single-channel track which can be panned normally for mixdown to a stereo bus (such as **Master Out**), and avoids wasting disk space on empty or duplicate audio tracks.

For further details, see manual section [4.1. Routing–Connections, Ports, Tracks and Buses](#).

How To - 4 Sample MIDI Composition Workflow

[How To - Contents](#)

Introduction

This How To introduces one possible approach to composing with MIDI tracks. It gives a brief outline of each stage of the process - namely track set-up, composition, recording, external routing and audio export. It's aimed mainly at beginners who aren't overly familiar with sequencers/DAWs, but some of the Qtractor-specific information may also be of interest to more experienced users. Note that a number of the tasks mentioned here are already detailed in the [wiki](#), so this How To contains several links to it in order to avoid repeating information.

Different people have different needs in terms of workflow, so this How To is by no means a definitive guide. However, it should hopefully provide some kind of direction for people who are uncertain where to start. Finally, it assumes that you already have Qtractor up and running with JACK etc, so if this is not the case, please check the set-up information in the wiki before continuing.

**** NOTE **** This guide was originally written for Qtractor V:0.5.x, when it was necessary to bounce any MIDI data to audio tracks in order to export it. However, it is now (since V:0.6.7) possible to export “internal” MIDI tracks (i.e. those which use plugin instruments, such as the xsynth example detailed below) directly as audio, thus avoiding the bounce step. Nevertheless, for “external” MIDI tracks (those which send data to external instruments driven by JACK-MIDI or ALSA-MIDI, the output of these instruments then being routed back into Qtractor), the below workflow still holds true. It follows that a MIDI session which has both internal and external tracks will necessitate a “hybrid” workflow, wherein you first bounce the external tracks to Qtractor audio tracks, then export these together with the internal tracks.

For internal tracks, the workflow described by this How To will have the following differences:

- **Section 1:** No need to create dedicated audio outputs or audio tracks
- **Sections 2, 3, 5:** Ignore
- **Section 6:** Create the Aux Sends in the MIDI tracks (note that they will still be *audio* Aux Sends, even though in MIDI tracks)

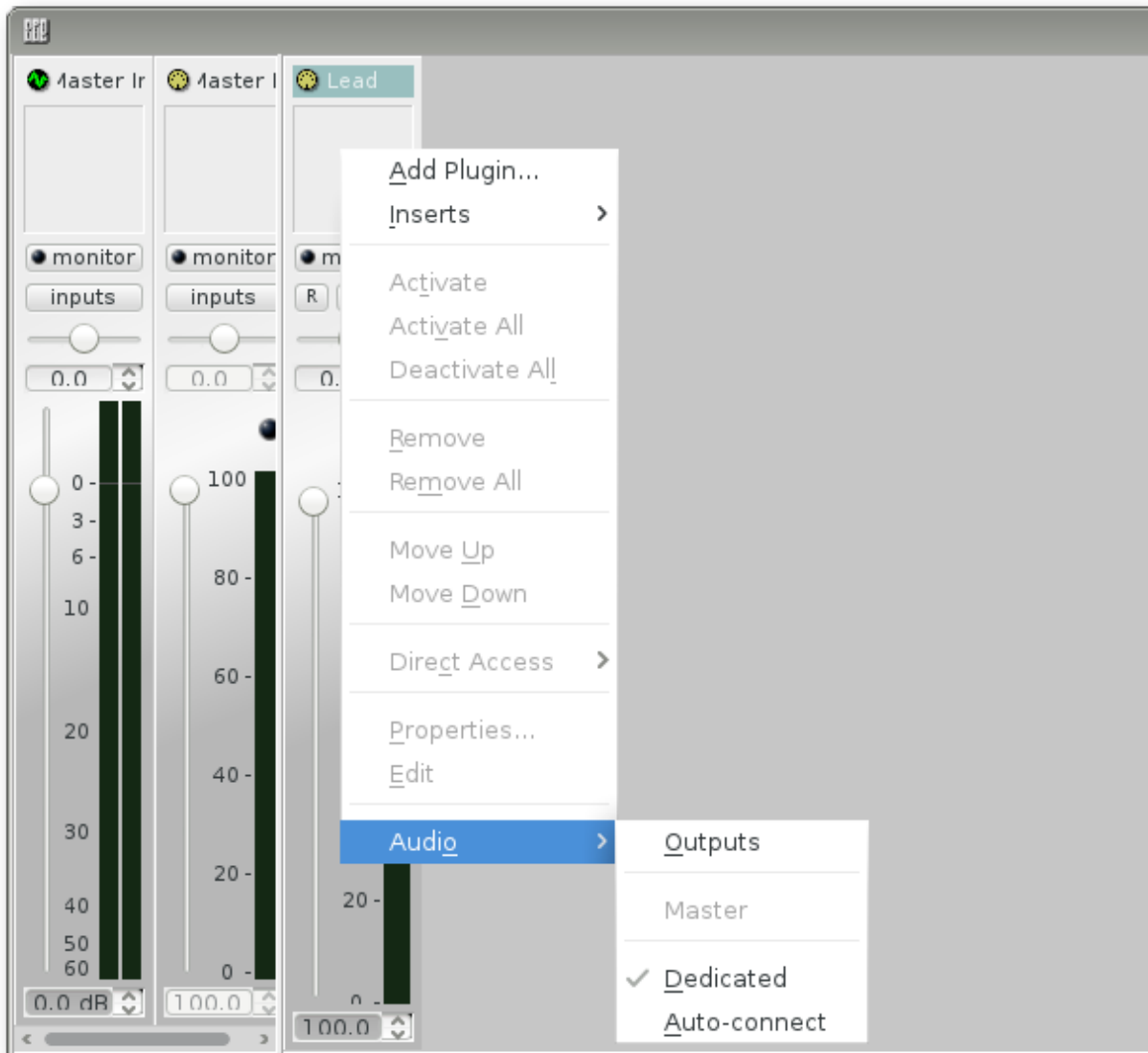
1. Creating Tracks

To keep things simple, we'll create just two MIDI tracks - Lead and Bass. Create a MIDI track, as detailed [here](#), naming it “Lead”. Then, open up the Mixer, by clicking on its icon.



The Mixer icon

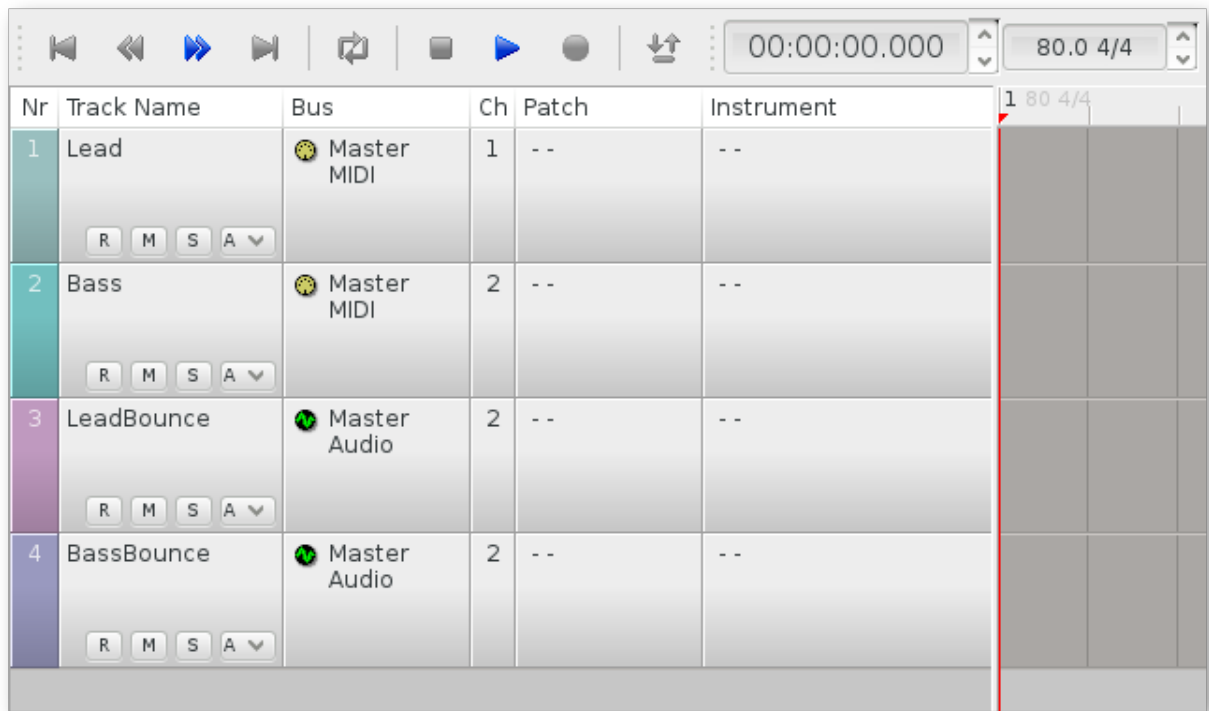
Right-click in the area directly under the “Lead” text, navigate to **Audio** and choose **Dedicated**. This will create an output specifically for this track, which we’ll later connect up. Note that you can choose to **Auto-connect** the output – doing so means that it will automatically be connected to `system:playback_1` and 2. Whether you should do this or not will depend on your set-up, but for the purposes of this How To, we can leave it enabled.



Creating a dedicated output

Then, create another MIDI track, naming it “Bass” and repeat the above.

Next, we’ll create two Audio tracks, to which we’ll later record, or “bounce”, the output of the MIDI tracks. Using the same Track window as you used for the MIDI tracks, this time, select **Audio**, instead of **MIDI**, under **Type** and create a track named “LeadBounce” (there’s no “dedicated output” step here). Once done, create another Audio track, called “BassBounce”. Qtractor should then look like this.

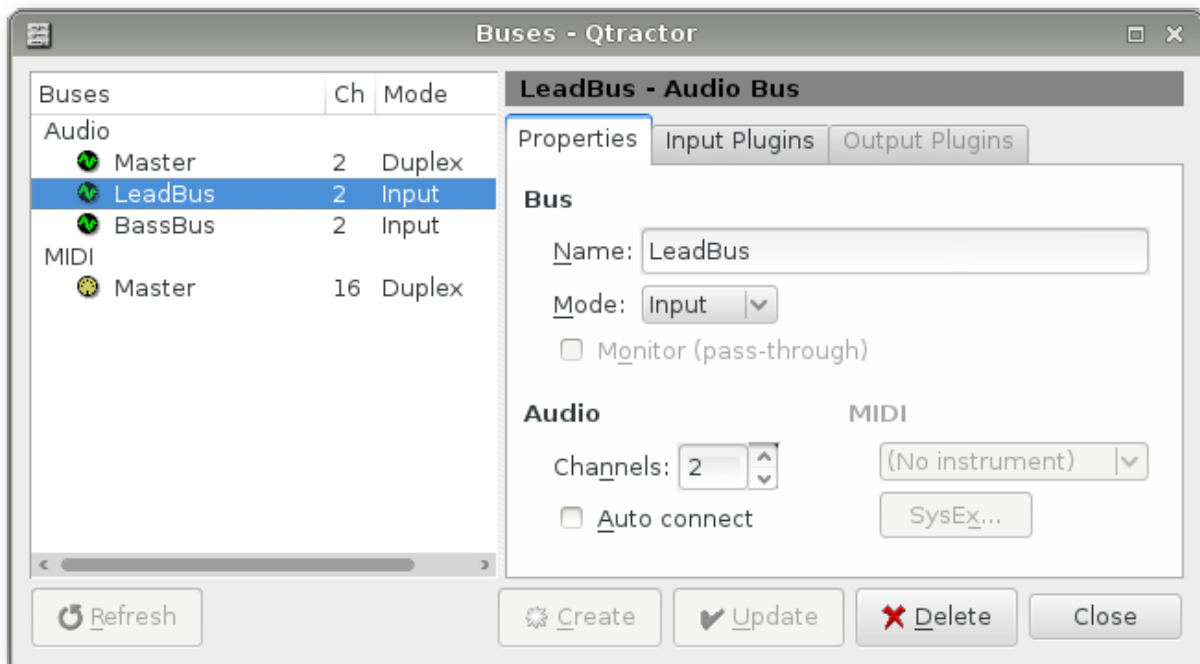


MIDI and corresponding Audio tracks

2. Creating Buses

In order to send the output of the MIDI tracks to their Audio tracks, we use **Buses**. Buses channel the output from a particular source to a particular destination. For more detailed information on them and on the concepts of routing, please see the [wiki](#).

The method of creating a Bus is covered in [this How To][How To - Create Individual MIDI and Audio Buses-Ports]. Using these instructions, create an **Audio** bus, with the **Mode** set to **Input**, calling it “LeadBus”. Then, create another in the same way, calling it “BassBus”. As with the MIDI tracks’ dedicated outputs, you can choose to Auto-connect or not (for the purposes of this How To, it doesn’t really matter). Your Bus window should then look like this.

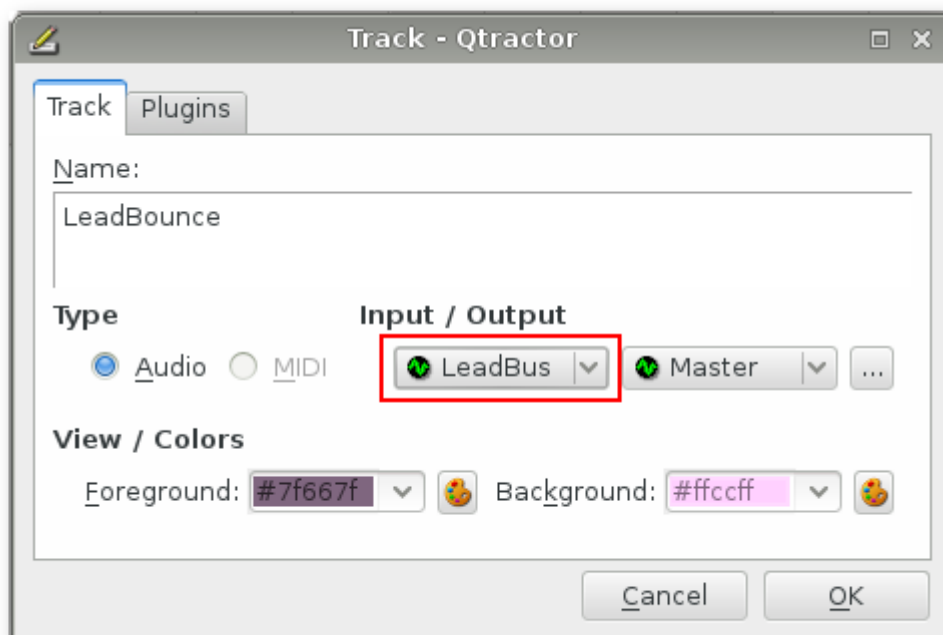


Creating Buses

Note that you can also create mono Buses to record into a mono audio track. To do this, you would set **Channels** to **1**, as covered in [this How To][How To - Set the number of channels an audio track records].

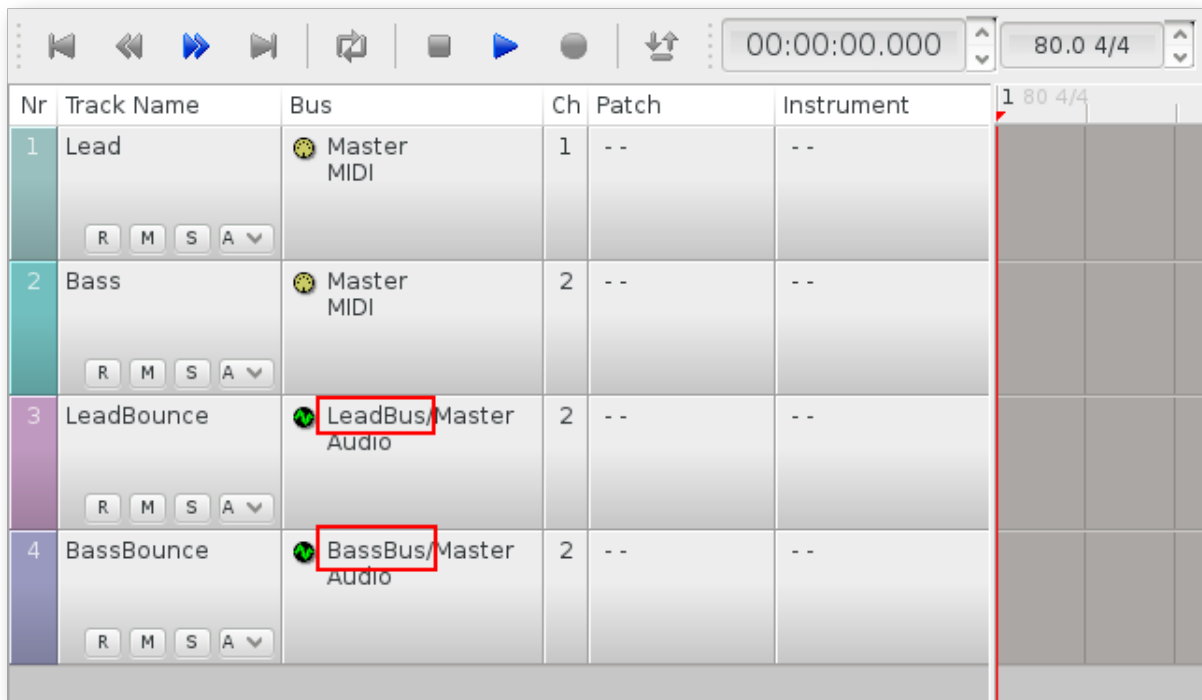
3. Connecting everything up

Although we now have MIDI tracks to compose in, Audio tracks to record in and Buses to route the audio, we won't be able to use them until we connect everything up correctly. Double-click on your **LeadBounce** track in the main view and, in the **Track** window which appears, set its Input to **LeadBus**.



Creating Audio track Inputs

Next, do likewise for **BassBounce**, connecting its Input to **BassBus**. Qtractor should then look like this.



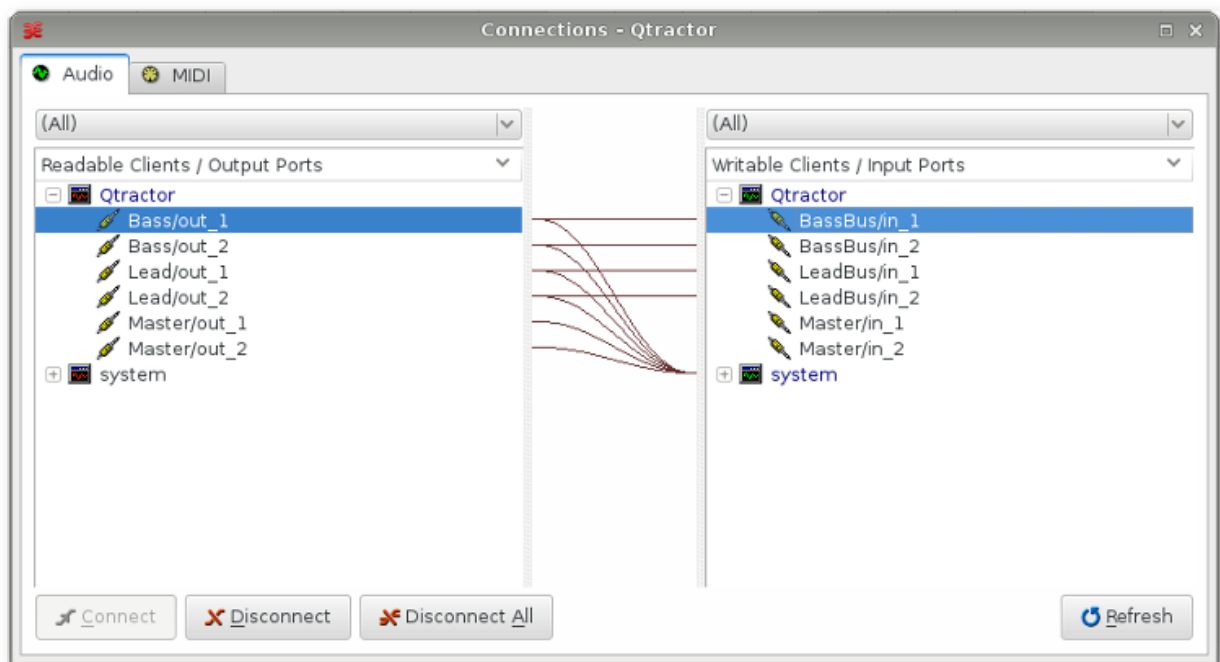
Audio tracks with their Inputs routed from Buses

The Audio tracks are now ready to receive input from their respective Buses, but we still need physically to *connect* the flow of audio. We do this via the **Connections** window. Click on its icon to open it.



The Connections icon

If you've used [QjackCtl](#) (also by the author of Qtractor) you'll be familiar with the layout. On The **Audio** tab, connect the Lead and Bass outputs on the left (which are our MIDI tracks' dedicated outputs) to their corresponding inputs on the right (which are our Buses). You can either drag the connections across, or highlight the source and destination and click **Connect**. The Connections window should then look something like this.

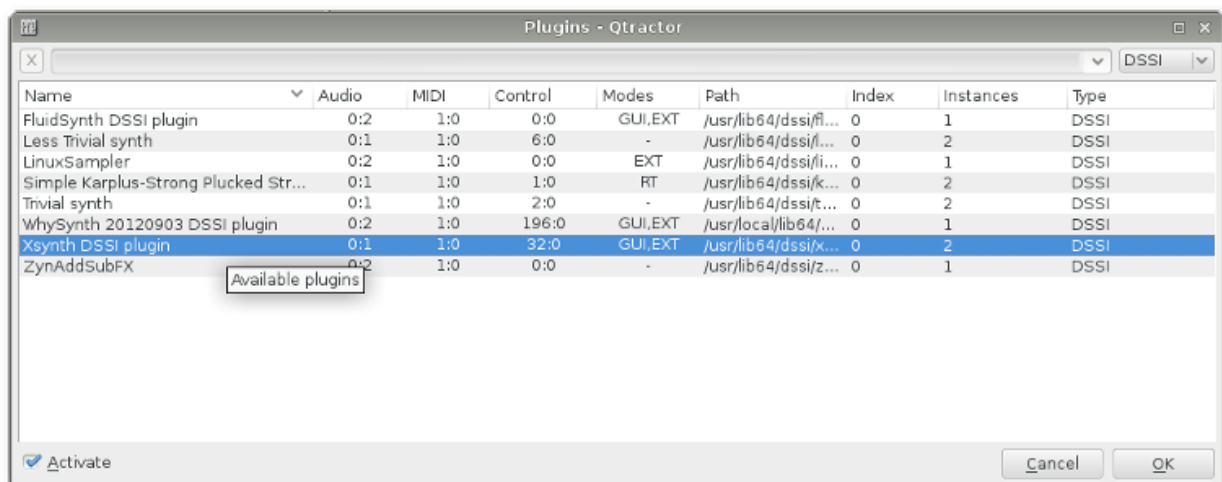


Connections window with Dedicated Outputs connected to Buses

Now, the output of the MIDI tracks will be routed to their respective Buses, which will, in turn, be routed to their respective Audio tracks to be recorded.

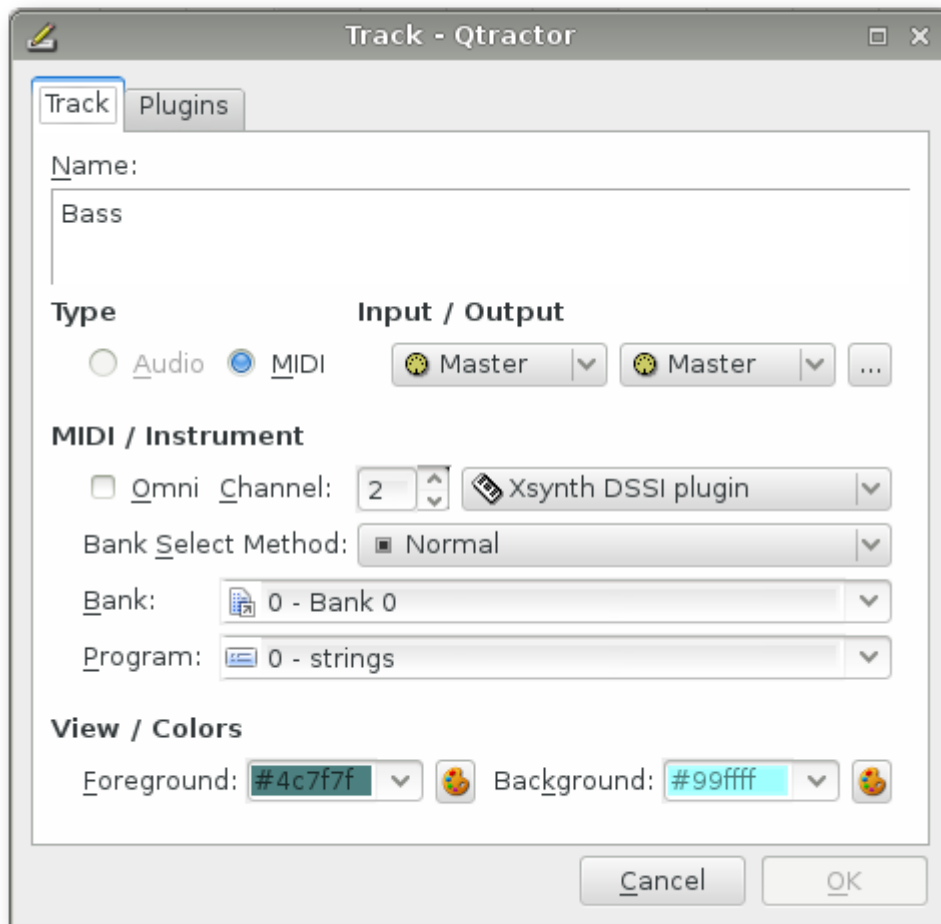
4. Compose!

Exactly how you go about composing will depend on your needs, but for the sake of this How To, we'll do things simply and use the soft synth, [xsynth](#) (though any other will do). Back in the Mixer window, right-click under the "Lead" text, as you did previously, but this time, choose **Add Plugin**. This will bring up the Plugins window. Here, you can select from among anything you have installed, of the various plugin types supported (LADSPA, DSSI, VST (Linux native) and LV2). For more details on these, see the [wiki](#). xsynth is a DSSI plugin, so resides in the DSSI tab.



The Plugins window

Choose the synth patch you want via xsynth's GUI and make sure that the plugin is enabled in Qtractor (green light next to its name in the Mixer). Then, double-click your MIDI track in the left pane of the main view and, in the **Track** window which appears, select “xsynth DSSI plugin” in the **MIDI / Instrument** area. The **Bank** and **Program** fields should then reflect whatever you chose in the synth's GUI and, once you click **OK**, this information should also appear in the main view. For DSSI plugins, you should be able to select patches via either the **Track** window or the plugin's GUI, with the other automatically updating, but this may not be the case with LV2 plugins. Please see the note in the [wiki](#) about this.



Track window with patch loaded

Once you've set up your synth for the “Lead” track as above, repeat the process for the “Bass” track.

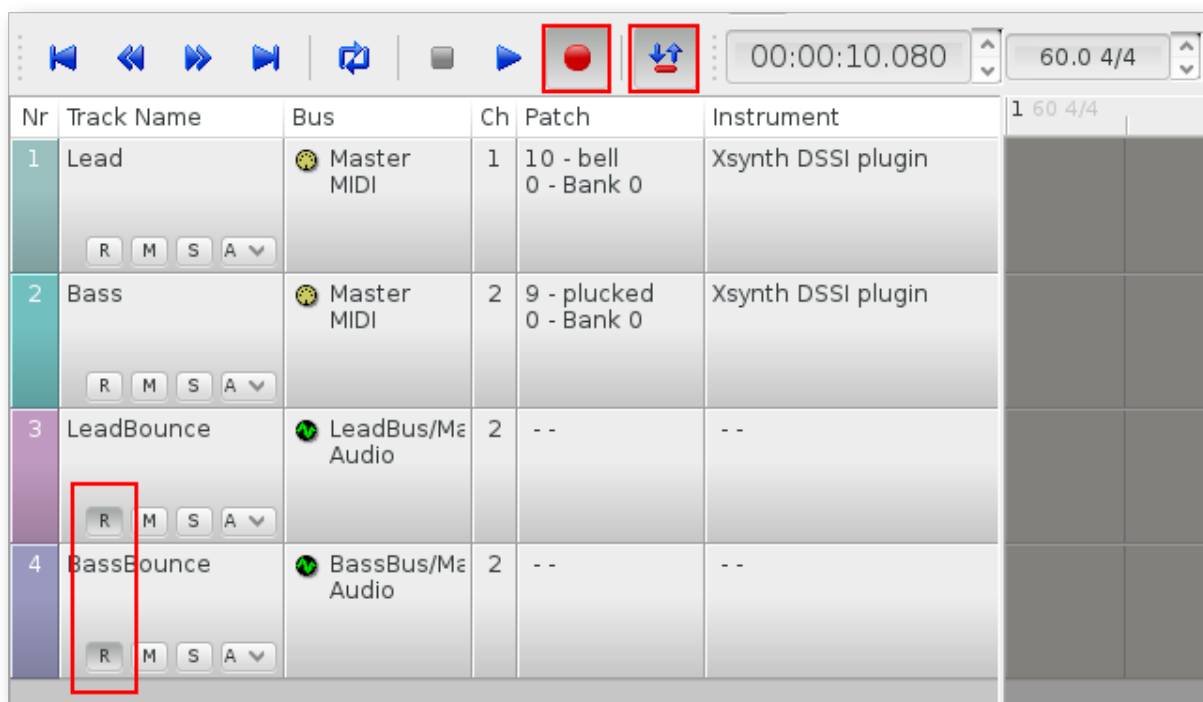
*Note that when working with soundfonts (for example, via the [FluidSynth DSSI](#), or [Calf Fluidsynth LV2](#), plugins), you can use the **View -> Instruments...** menu to import their patch names. To do so, click the **Import** button and select the desired **.sf2/.sf3** file; the patch names will then be imported by Qtractor.

You can also use an external synth, rather than a plugin instrument. If you choose to go down this route, you would use the **Connections** window to route the MIDI track's output bus (the default is *Master*) to your synth and, if necessary, set a specific channel in the **Track** window described above. Then, again in **Connections**, you would route the synth's audio output to your Qtractor audio track's input bus (in the context of this How To, either **LeadBus** or **BassBus**). For more details on using external synths, see the [wiki](#).

Once your synths are set up for both tracks, you can use whatever means you like to compose. You may choose to use an external [MIDI controller](#) to play in your performance in real-time, or input notes manually via the [MIDI Editor](#).

5. Recording

Once you're happy with your MIDI composition, it's time to record it as audio. Arm the Audio tracks for recording ("R" button on the track, via either the main view or the Mixer), move the playhead (red vertical line) to somewhere before your MIDI composition starts, enable the red "Record" icon, then click the "Play" icon to start the transport rolling (and the "Stop" icon to stop). You can also use the Punch in/out facility to specify where recording starts and ends, setting these points with the Edit-head and Edit-tail markers (blue vertical lines).



Qtractor set up to record, with Record and Punch in/out icons highlighted

If you've done everything correctly, once you start the transport, the signal from the MIDI tracks should be routed via the Buses and be recorded in the corresponding Audio tracks. During the bouncing process there's always a chance that x-runs can occur, adversely affecting your audio file, so it's best to check for this. The precise positions of x-runs are unfortunately not marked visually in the track area, but when one occurs a red warning indicator will appear in the session state information at the bottom-right of the screen. X-runs are also logged, together with their time of occurrence, in the Messages pane at the bottom of the screen, so use this as a reference. Once you have your audio safely recorded, you can adjust gain levels, panning etc via the Mixer.



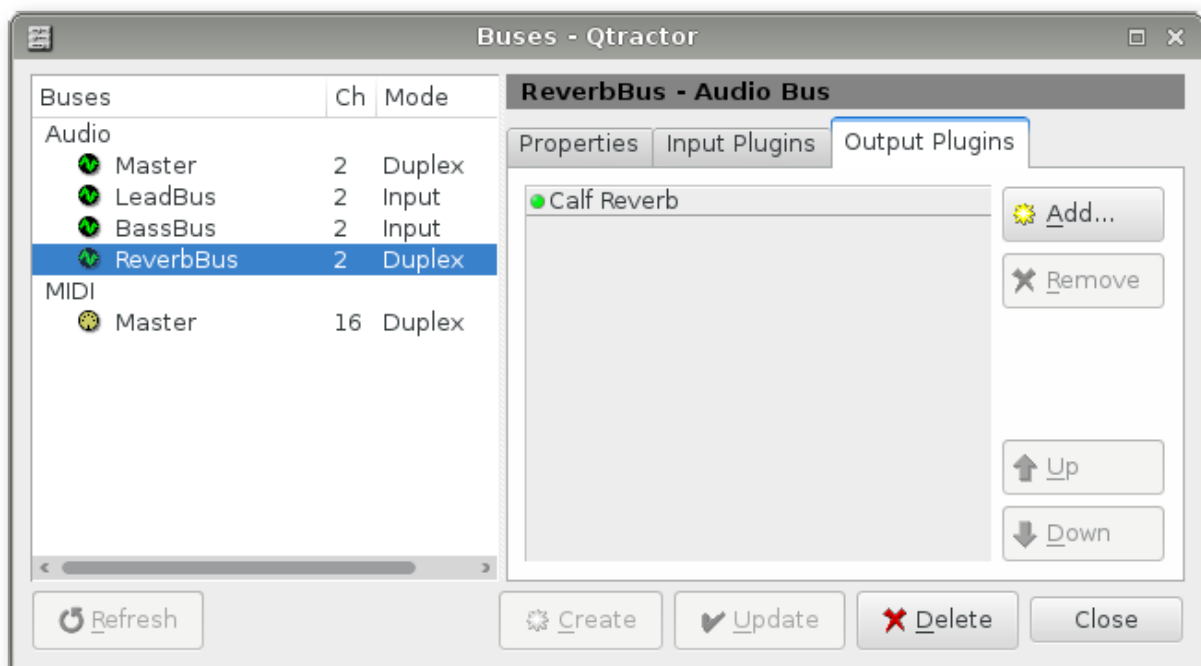
Session state information showing x-run indicator

When playing back your audio tracks, note that you may, depending on your set-up, have to mute the MIDI tracks (using the "M" button), to avoid the audio output doubling up.

6. Effects plugins and Aux Sends

In most cases, you'll want to add some effects to your tracks. This can be done very easily via the **Add Plugin** menu of the Mixer, as covered in 4. *Compose!* above. You can also add several plugins to the same mixer strip if desired; they are processed in order from top to bottom. If you need to change the order of an already-inserted plugin, you can simply drag and drop it.

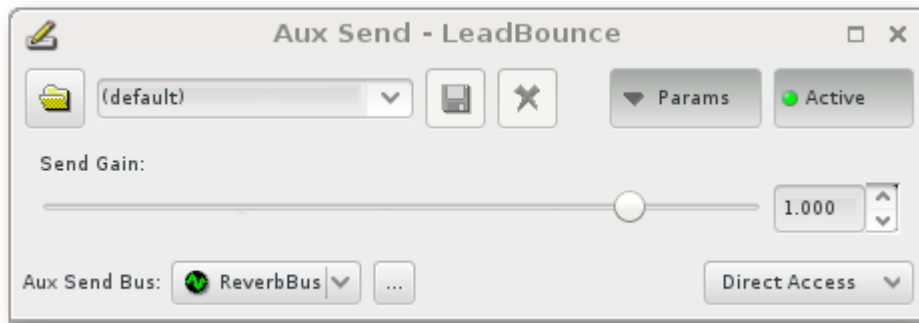
You may also want to send *several* tracks to the *same instance* of a plugin in order to share it. This is a reasonably common requirement, particularly for reverb, and is catered for in Qtractor with **Aux Sends**. Although we have only two tracks here, to illustrate the theory, we'll use Aux Sends to send the two tracks to another Bus, so that they can share a reverb plugin. First, create a **Duplex** Audio Bus via the Bus window we used earlier, calling it "ReverbBus". In the **Output Plugins** tab, right-click in the blank area, choose **Add Plugin**, then select the reverb plugin you want to use, making sure it's active (green light). You can also do this via the Mixer. If you're not sure which plugin to use, the [Calf LV2 set](#) is a decent place to start.



Creating a reverb Bus

You'll probably also need to connect ReverbBus' outputs to *system:playback_1* and 2, via the **Connections** window we used earlier, so that you can hear the reverb effect.

Next, in the Mixer, right-click under the Audio track name "LeadBounce" and choose **Inserts -> Audio -> Add Aux Send**. In the window which appears, set the Aux Send to "Active" (top right button), choose **ReverbBus** in the **Aux Send Bus** combo box and choose the amount you want to send via the **Send Gain** slider. Then, do the same for "BassBounce".



The Aux Send window

What value to set **Send Gain** to depends on exactly what you want to do and how you have your reverb plugin set up, but the basic theory is to have an “overall” setting in the reverb plugin itself, then adjust the amount you send to this plugin via each track’s Send Gain slider. In this way, you can control the amount of reverb applied, in order, for example, to mimic an instrument’s placement on the stage. Use a combination of Mixer levels, plugin settings and Aux Send gain amounts to achieve the effect you want.

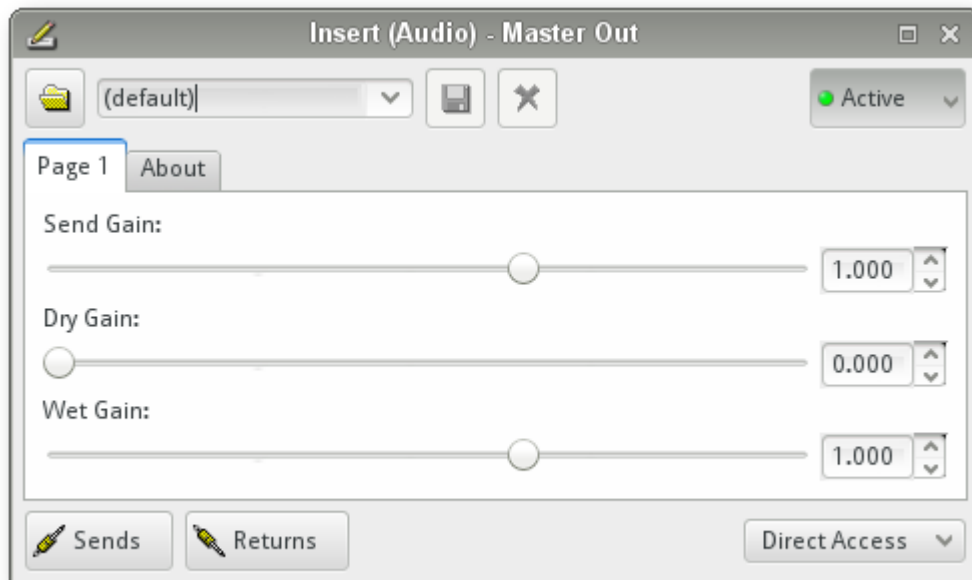
There are a couple of things to be aware of with regard to the way Qtractor routes audio when using Aux Sends:

- **Pre/post-fader placement:** All plugins in Qtractor are *pre-fader* - there is no facility to place them post-fader. This can cause issues in cases such as the above when you want to send an audio track to a reverb bus, as you would generally (though, again, it depends on what you want to do) place the send post-fader: the reason for this is that with the send in the post-fader position, the dry/wet (non-reverb/reverb) ratio is maintained even if the fader is adjusted. This limitation can be mitigated to an extent by inserting any kind of plugin which controls gain *before* the Aux Send and using that, rather than the Mixer’s gain fader, to control the track’s gain
- **Panning:** If you manipulate the Mixer’s pan setting with an Aux Send enabled, you’ll only be panning the original signal; the “sent” signal will enter its destination Bus panned to the centre. This means that you’ll hear the original signal wherever you’ve panned it, but the Bus’ copy in the centre (unless you’ve panned it again there). If you want to pan both the signals in tandem, add a plugin which controls panning before the Aux Send and use this, rather than the Mixer’s pan setting, to pan the track

7. External routing using Inserts

In Qtractor, **Aux Sends** are used for routing audio internally, as described above. **Inserts** are similar, but are used to route audio externally. Once you have your mix set up, you may want to route it to an external source, such as [JAMin](#). The details of how to use JAMin are outside the scope of this How To, but we will cover one way of connecting it up to Qtractor. The [manual](#) features some information on how to use the program if you’re unfamiliar with it.

With JAMin running, in Qtractor’s Mixer, right-click below the name of the **Master Out** Bus (right hand side) and choose **Inserts -> Audio -> Add Insert**. In the window which appears, set the Insert to “Active” (top right button). Leave **Send Gain** at 1.000, **Dry Gain** at 0.000 and **Wet Gain** at 1.000, unless you have some other set-up in mind.



The Insert window

Next, use the **Sends** button to connect the Insert to JAMin and the **Returns** button to connect JAMin back to the Insert. These buttons open up the same **Connections** window we used earlier, so this process should be straightforward. Once you’ve connected everything up, repeat the above process for **ReverbBus**, but connect *only* the **Sends** (leave **Returns** unconnected). ReverbBus will now have two entries in its Mixer strip – the reverb plugin above and the Insert below. This means that the reverb plugin will be processed first, followed by the Insert, which is the order we want.

The audio from both your Buses should now be routed into JAMin - where you can use its many features to fine tune your mix - then back into Qtractor’s **Master Out**. In this way, the Master Out strip of the Mixer, at the point below the JAMin Insert, will have a “complete” stereo signal, containing the output of both buses. Beyond (that is, below) this point, you can perform any processing/analysis relevant to the mix as a whole, such as goniometer and mono checks.

Note that upon connecting JAMin up, you may find that it is also sending its output directly to `system:playback_1` and `2`, which will cause some unpleasant doubling up of audio. To fix this, disable the outputs in JAMin, via* **Ports** -> **Out** -> **out_L** & **out_R**.*

There’s an important point to be aware of when working with Inserts and other external connections: Qtractor will only restore such connections *if they are present when the session is saved*. For example, if you proceed as above, save, then close and restart both Qtractor and JAMin, the connection will be correctly restored. However, if you close JAMin first, make a change in Qtractor and save (i.e. with Qtractor/JAMin in a “non-connected” state), upon restarting Qtractor/JAMin, the connection will be lost and you’ll need to make it again.

This behaviour can be avoided by always ensuring that you have your external connections in order before closing Qtractor (easier said than done when using many external applications). It can also be mitigated somewhat by using QjackCtl’s *Patchbay* feature to make the connections persistent; however, this too is rather unwieldy as it means adding each item manually to the Patchbay. A third approach would be to use a session manager, such as QJackCtl’s *Session* feature (which employs “JACK session”), or [NSM](#).

8. Exporting

We've now covered how to set up MIDI tracks, bounce their audio to Audio tracks, add plugins and route the mix to JAMin. The only thing that remains is to export the result into a single audio file. This can be done by right-clicking within the left pane of the main view and choosing **Export Tracks -> Audio....** For details on the options available, please see the [wiki](#). In the context of this How To, we'd highlight both **Master** and **ReverbBus** in order to export them together.

How To - 5 Sidechain Workflow with Carla Plugin

[How To - Contents](#)

Note: This article is obsolete. Since Qtractor 1.5.7, auxiliary sends allow you to route audio signals from channels through a matrix. Therefore, external plugins are no longer necessary for this task. For example, if you want a sidechain effect in a stereo environment:

1. Create a four-channel SideChain bus that houses the sidechain processing plugin and an auxiliary send to the destination bus in the mix.
2. Send the signal to be processed to channels 1 and 2 of the SideChain bus with an auxiliary send.
3. Send the modifier signal to channels 3 and 4 of the SideChain bus with another auxiliary send.

This will allow you to mix the original signal with the processed signal in a flexible and powerful way. In any case, the article is still useful for understanding what a “sidechain” is, its practical uses, and examples of various workflows.

- **Author:** G3N-es.
- **Difficulty level:** Advanced.
- **It is assumed that you already know:**
 - Configure Jack.
 - Configure Qtractor (plugin paths, etc).
 - Create a project with midi and audio tracks.
 - General notions of mixing and mastering.
 - Configure midi and audio tracks.
 - Load plugins on tracks and buses.
 - Install Plugins.
- **You will learn to:**
 - Create advanced audio routing.
 - Take advantage of that knowledge to create sidechain flows.

0. Purpose of the tutorial

Have sufficient knowledge to be able to apply different workflows with sendings to sidechains in Qtractor. The workflows proposed here are only a partial reference of infinite possibilities. Related content: [How To - 6 Alternative Sidechain Workflow with Aux Sends](#)

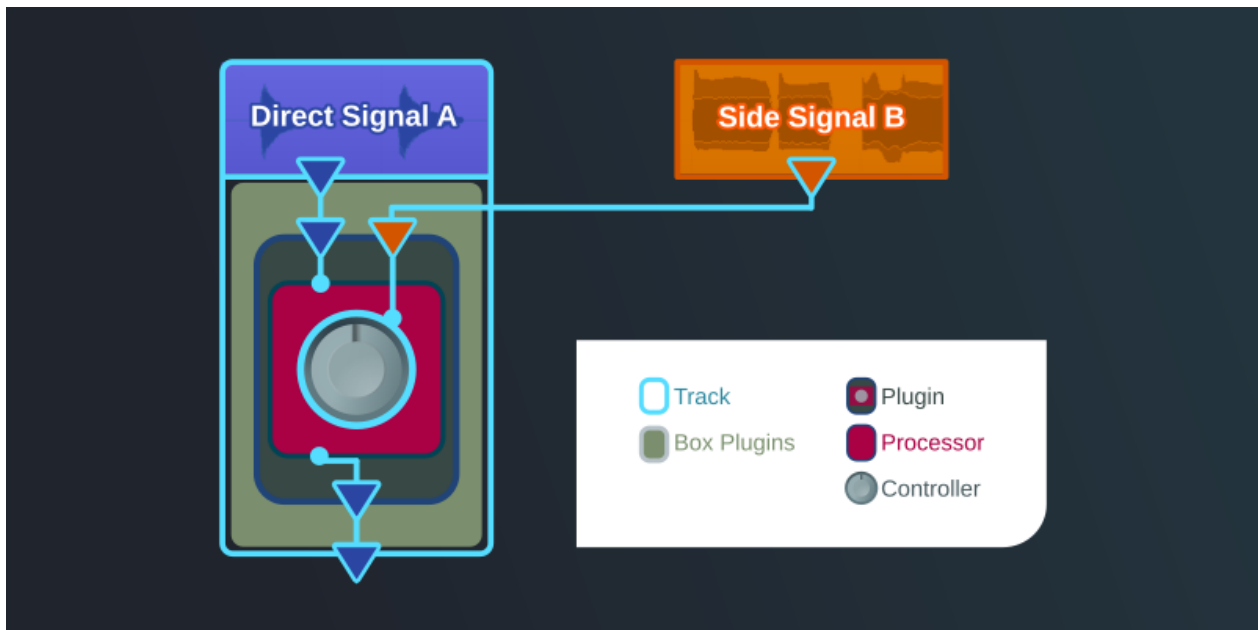
Qtractor does not have pre-built workflows. In exchange, it offers you total freedom to create your own workflows.

0.1. Ingredients

- Qtractor 0.9.36 or higher
- Plugins lv2
 - [GM Synth](#)
 - [Carla- lv2](#)
 - [Calf-plugins](#)

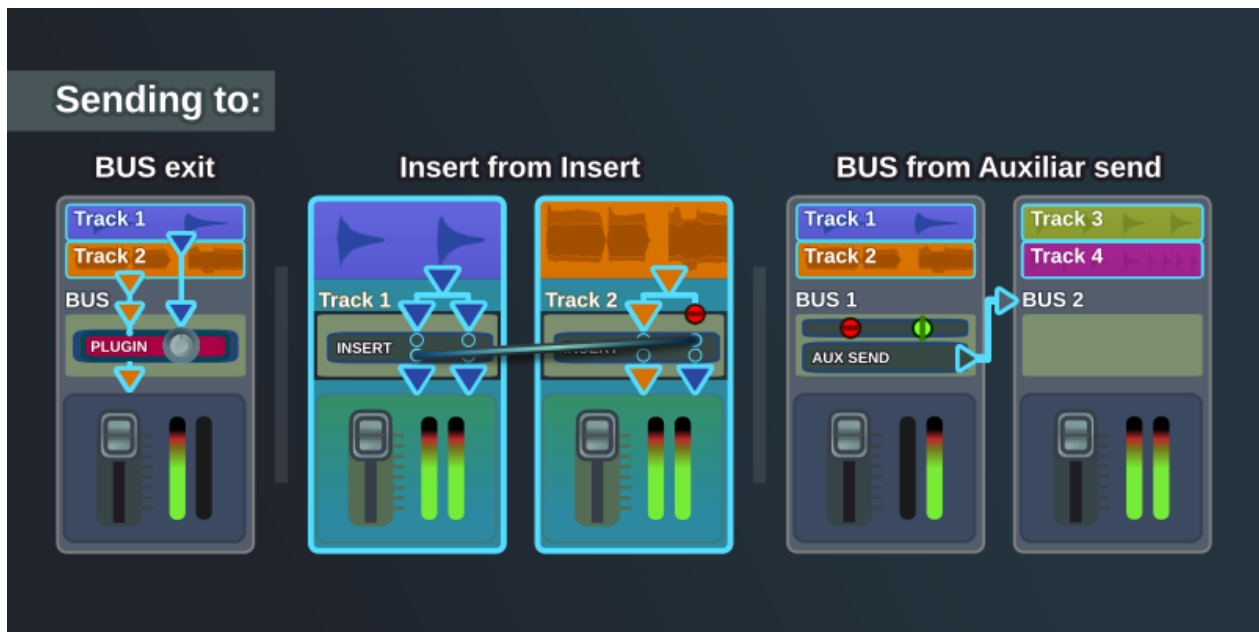
0.2. What is a “sidechain”?

Sidechain is a signal sending technique that allows modifying or controlling an audio signal “**direct A**”, with another audio signal “**external B**”.

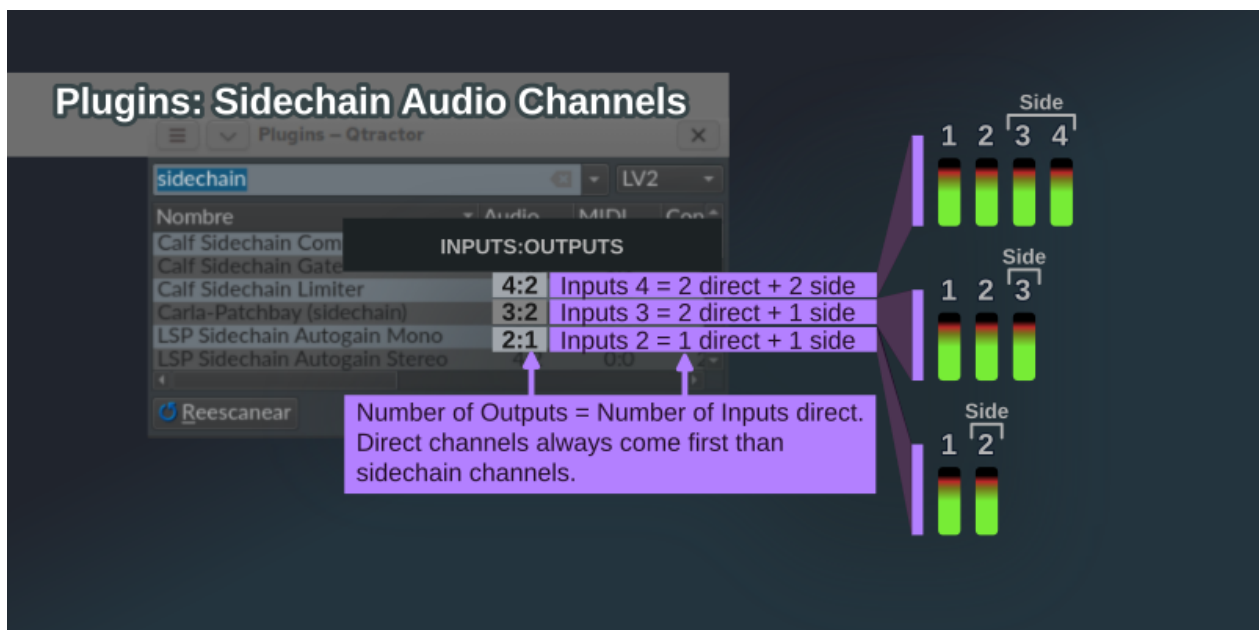


Types of sendings in Qtractor

- Sending to own bus by default
- Sending to track (inserts)
- Sending to auxiliary bus



Plugins with sidechain They are characterized by having more inputs than outputs: Every sidechain plugin must have extra input/s for the sidechain “**external B**” signal in addition to “**direct A**” input/s.



“**Direct inputs**” are the first ones that correspond in number to the outputs. The remaining ones are “**entries for sidechain (or external)**”. The workflow is conditioned by the demands of each specific plugin (mono, stereo, purpose of the plugin, etc.).

Two very widespread uses:

- Dynamic sound control (compressors, expanders...)
- Vocoder synthesizers.

However, there are no limits, all types of plugins can be developed with a sidechain, where an “**external B**” signal “automates” a certain parameter or parameters of a plugin in order to control or modify the “**direct A**” signal. Not all plugins support sidechains, they must have been designed and programmed expressly for this purpose.

0.3. Channel behavior in Qtractor

1. Undefined channels. A bus is not mono, stereo... A bus simply has channels. It is you who, through the interconnections, define the configuration (mono, stereo, surround... with or without sidechain). The number of channels on a Bus can be modified at any time, although it is advisable to design the project connections and workflow in advance.

3. The buses determine the number of channels in the tracks. Sidechain plugins have extra inputs. So we need extra channels that allow the signal to be directed towards them. Qtractor buses allow you to create these new channels.

4. Although buses (and therefore tracks) with different numbers of channels can coexist, it is not recommended. If you plan to work with sidechain, buses with the same number of channels are the best option. Otherwise, “Auxiliary Sends” will not work and “Inserts” may create conflicts.

5. An audio track will attempt to inject audio to all available channels before entering the plugin chain. This allows you to combine mono and stereo audio files into a single track in an intuitive and simple way. In general it is desirable behavior. It is not in the specific case of sidechains. However, it is easily solved with plugins that mute or interrupt channels.

Thanks to the “**switch plugin**”, we now have the 2nd channel free to be able to carry out sidechain workflows. What at first seemed like a configuration of stereo track, it turned out to be a mono + sidechain channel configuration. I want to emphasize this idea, although the tracks start from a stereo configuration for panning control, a Qtractor Bus gives you undefined channels, so that you define them. And this feature is what we are going to take advantage of to apply sidechain.

6. The “midi instruments” generate an audio signal within the plugin rack. They do not behave like an audio track. They only occupy the bus channels that correspond to their outputs. That is why with them it is not necessary to use switches. channels.

7. Standard sidechain channel configurations:

- **Mono:** 2 channels (the first for direct, the second sidechain).
- **Stereo:** 4 channels (first 2 for direct, remaining 2 for sidechain).

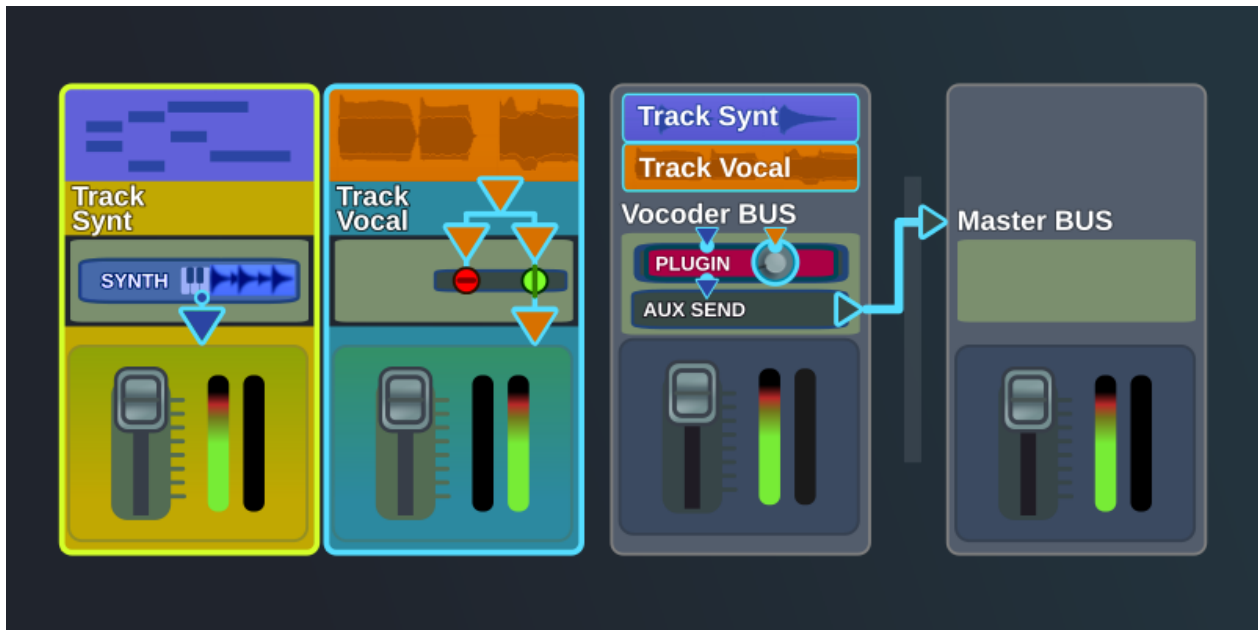
1. Send to own bus: Assemble a simple vocoder

Compatible with “**Jack (genuine)**” and “**PipeWire**”

It's the simplest way. There is no need to create new sendings. We take advantage of the signal send from a **track** to its default **bus**.

It could be summarized as, whenever we need a sidechain, creating a specific bus for all the tracks involved.

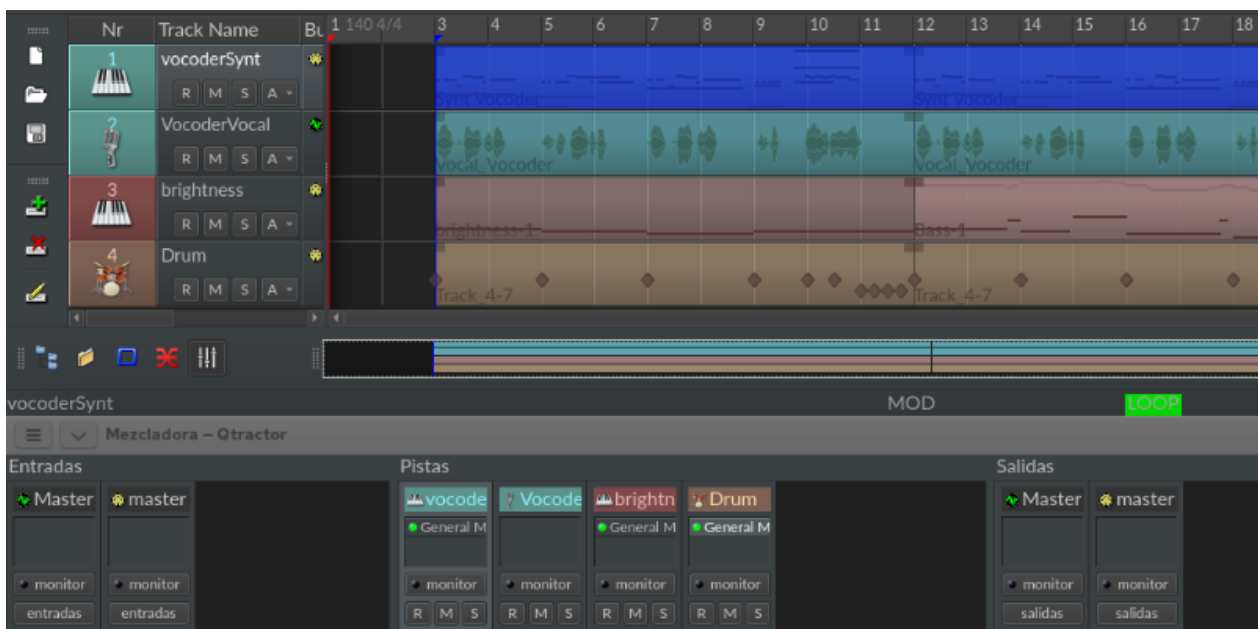
1.1 Connection scheme



1.2 From scheme to practice

EXAMPLE SESSION

- **vocoderSynt**: MIDI track responsible for providing the tonality (carrier) to the vocoder.
- **vocoderVocal**: Audio track responsible for providing the voice formant (modulator) to the vocoder.



CONFIGURE BUSES

The first thing we will do is create an exclusive **bus** for the “vocoder” tracks, where we will host the “Calf Vocoder” plugin.

In the mixer:

1. Double click on the **bus Master** title to access the **buses** configuration. (2) Rename it **“Vocoder”**, (3) in **“Channels”** assign **“4”**, (4) uncheck the **“Autoconnect”** box and finally (5) confirm the configuration with **“Update”**.

Now we will need a new Master channel.

1. Rename it as Master, (2) **“Mode”** change it to **“Output”**, activate the box (3) **“Autoconnect”** and click on (4) **“Create”**.



It's Qtractor's own and unique 2-step (Configure > Save) way of managing "buses". Therefore feels strange at first. However, once you see the logic, it is more productive than the traditional system (New > Configure > Save).

Adds the "Calf Vocoder" plugin to the "Vocoder" bus.

"Calf Vocoder" has a stereo sound concept, and arranges the channels as follows:



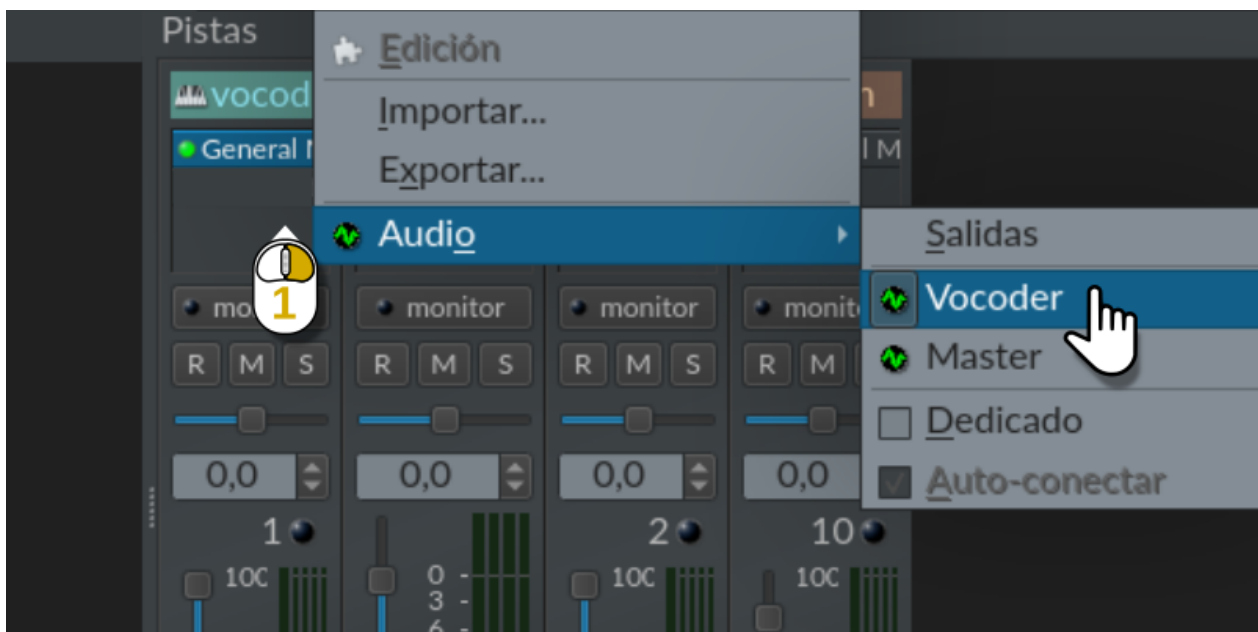
- The first two (direct signal): are for the **Carrier**, in charge of providing the tone.
- The remaining two (sidechain signal): are for the **Modulator**, responsible for providing the formant (which allows it to be perceived as a vocal timbre instrument).

Adds an “Aux Send” to the “Master” bus.

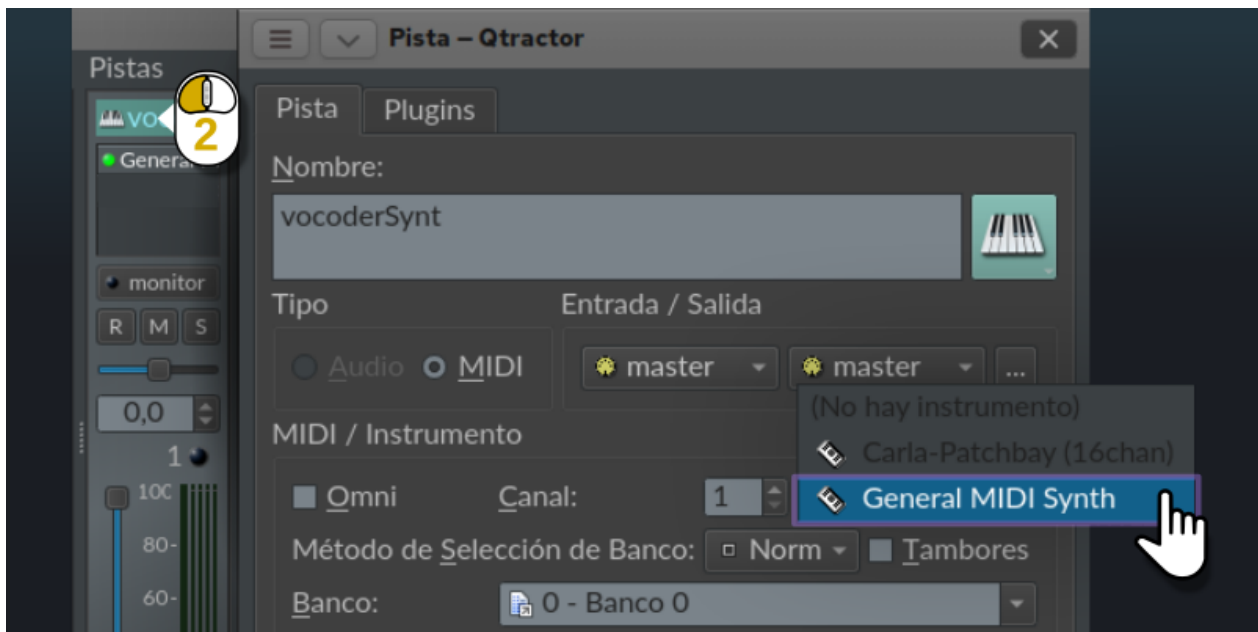
SET TRACKS

Track vocoderSynt:

1. Right click on the plugins box of the **midi track “vocoderSynt”**.
2. Assign as audio output bus: audio > Vocoder.



1. To set the timbre of “**General Midi Synth**”, go to the track properties (in the mixer, double click over the track title). Select “**General Midi Synth**”. Establish a bank and program of your preference.



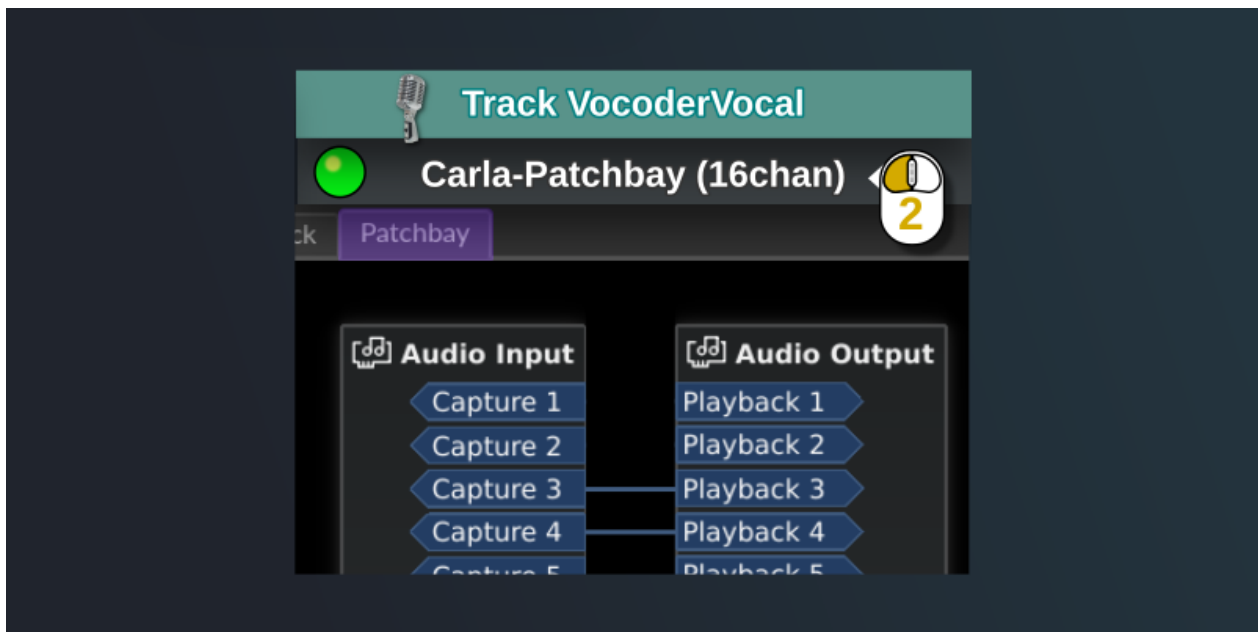
“General Midi Synth” can be deconfigured by playing “Piano default” instead of the program assigned by you. If this happens, reassign the bank and program. It is not very common, but it can happen.

Track VocoderVocal:

1. Double click on the title of the **audio track “VocoderVocal”**. Assigns the “Vocoder” audio bus as **output**.



1. Add the plugin “**Carla-Patchbay (16chan)**” connecting only **channels 3 and 4** intended for the modulator. We are going to use it as a switch plugin. It may seem that using Carla for this purpose is killing mosquitoes with cannon fire, however I have not found another that does the job better.



We already have all our connections!!!

Try and experiment with different settings on the Calf Vocoder and with different timbres on the carrier.

2. Send to track (inserts): Side compression

“PipeWire” Compatible: Tested in version 0.3.84.

“Jack (genuine)” Personally I haven’t had any problems. However, some users comment on the following incidents: * Inserts may not send audio. * Inserts may stop working when you reopen the session.

WARNING: Qtractor never had a tool for send to track. Using Inserts for this purpose is discouraged; they were not designed for this. However, at the time this tutorial was written, it was the only way to accomplish this task. Today, we have plugins that efficiently make sends to tracks. We recommend the LSP “Send” and “Return” plugins, hosted on the “Carla-Patchbay (16chan)” plugin, to reliably recreate this section of the tutorial. More information at [How To - 13 Make Successful Audio Connections](#)

2.1 Connection scheme



2.2 From scheme to practice

This time we will add the plugins from the sequencer instead of the mixer. The reason is simply educational, to see the many options that the Qtractor UI offers.

We will add a Sidechain Compressor to the track **“brightness” (intended for accompaniment)**. The intention is to **lose presence (volume)** when the **Vocoder (lead voice)** plays. This way we will achieve a cleaner mixture.

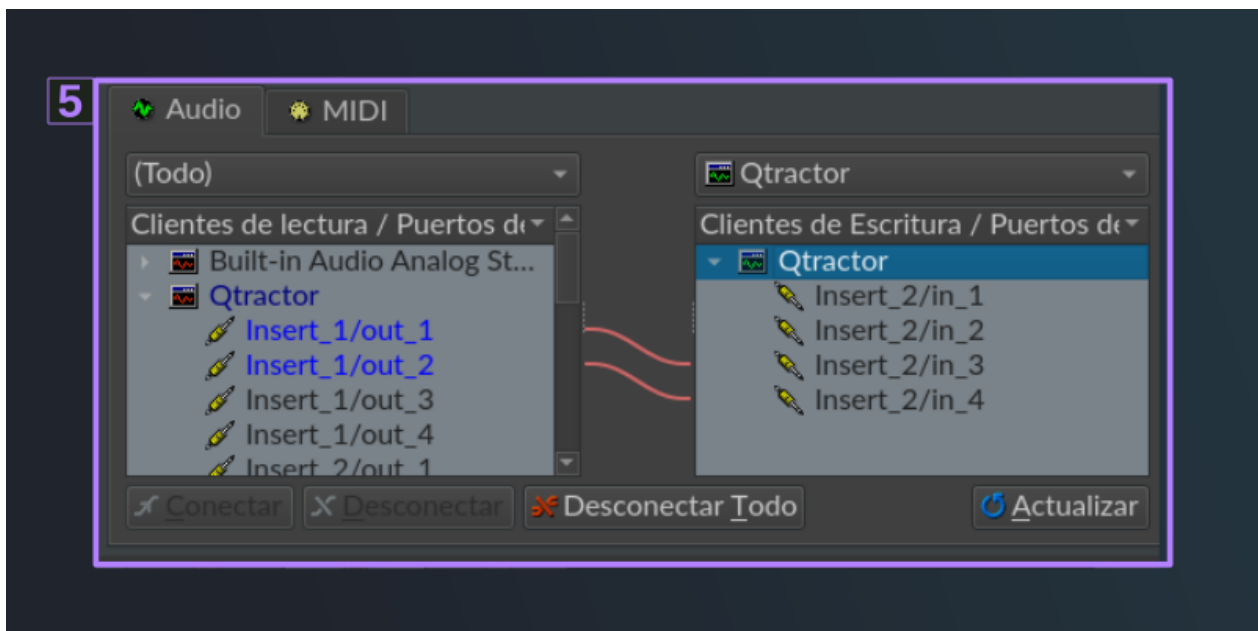
1. We added the plugin **“Calf Sidechain Compressor”** in the track **“brightness”**.
2. We added an **“Insert”** audio on the **midi track “vocoderSynth”**.
3. We added an **“Insert”** audio on the **midi track “brightness”** before **“Calf Sidechain Compressor”**.

We thus have an **“Insert 1”** for sending and an **“Insert 2”** for signal reception.



IMPORTANT!!! In Qtractor, inserts with a sending function must always have a lower number than those for receiving.

1. We right click on **“Insert 2”** and select: **Inserts > Audio > Returns**.
2. We connect outputs 1 and 2 of **“Insert 1”**, to inputs 3 and 4 of **“Insert 2”**.



1. Open **“Calf Sidechain Compressor”** and activate the sidechain. To ensure that it is working correctly you can activate the sidechain listener, and deactivate it after approval.



We already have all our connections!!!

Try and experiment with different configurations in **“Calf Sidechain Compressor”**.

How To - 6 Alternative Sidechain Workflow with Aux Sends

[How To - Contents](#)

Note: With Qtractor 1.5.7 all of the matrix stuff with plugins is no longer needed because Qtractor's AUX Sends have a builtin I/O matrix.

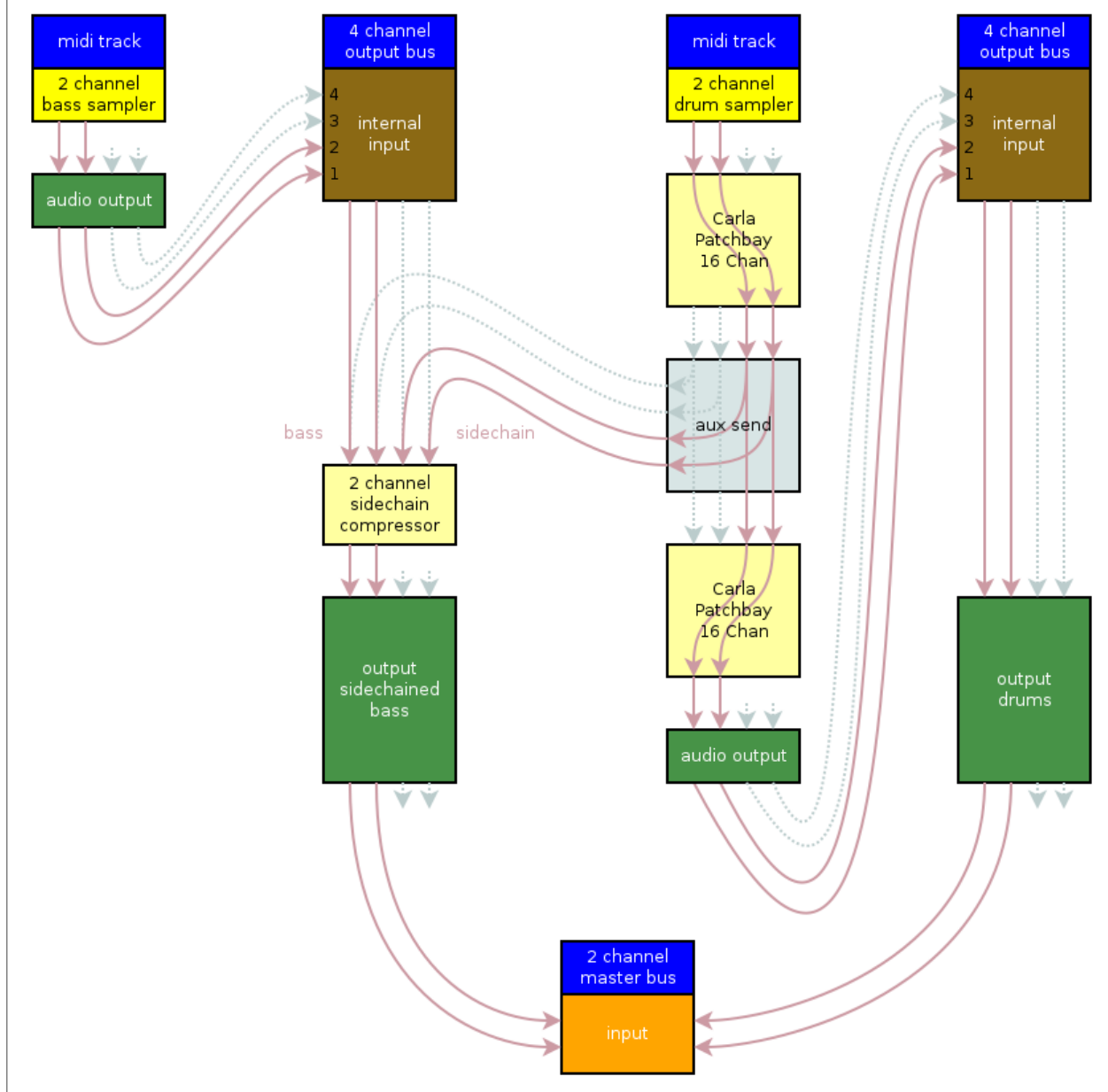
Alternative Sidechain Workflow with Aux Sends

Internal connections ensure low latency and tight timing as described in [How To - 7 Equal Latency for Tracks and Buses](#). This is also desirable for sidechaining. The sidechain signal should be synchronized.

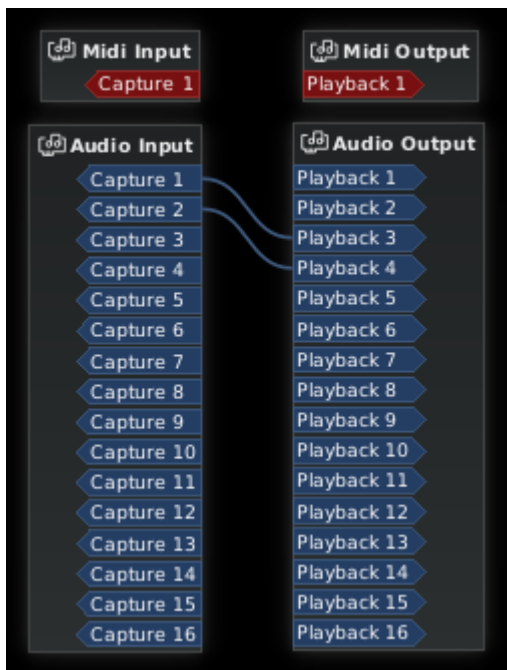
With aux sends, you cannot wire the channels individually. Most stereo sidechain compressors expect the signal to be compressed/ducked on channels 1+2 and the sidechain signal on channels 3+4. The channels of the sidechain must therefore be rearranged: Input channels 1+2 must be assigned to output channels 3+4, while output channels 1+2 should be quiet. [Carla Patchbay 16 Chan plugin](#) or [x42 Matrix Mixer](#) can do this. After using the aux send, we have to reverse the assignment, as the 2-channel master bus only uses channels 1+2.

Warning: At the time of this writing Qtractor may make a continuous sound after stopping/pausing playback when using Carla Patchbay 16 Chan plugin to rearrange channels. The volume seems to be proportional to the amount of compression/ducking at the playhead position. So keep the volume low while experimenting with this feature.

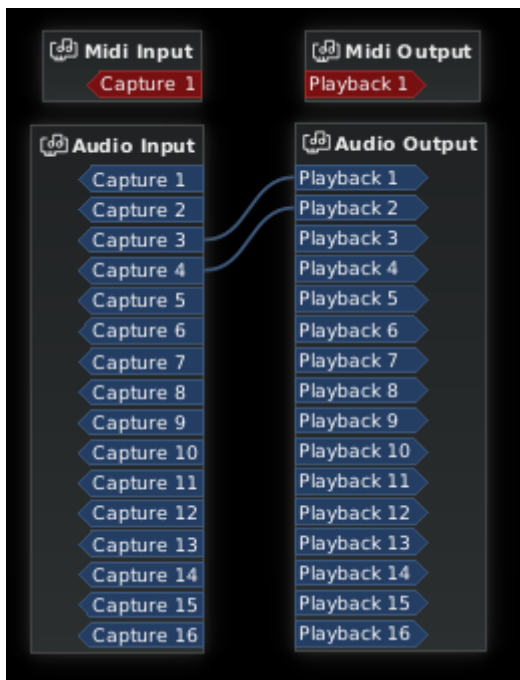
Qtractor sidechain with aux send



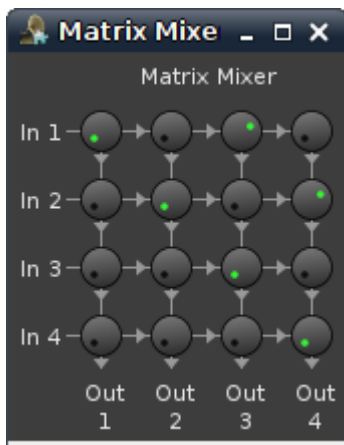
Patchbay view in Carla Patchbay 16 Chan plugin with input 1+2 routed to output 3+4:



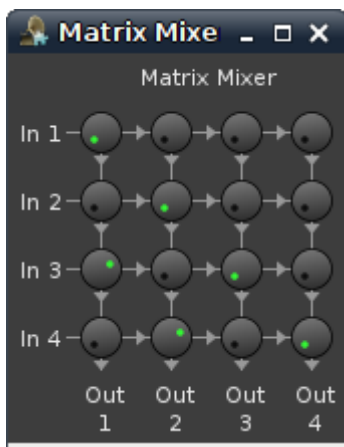
Patchbay view in Carla Patchbay 16 Chan plugin with input 3+4 routed to output 1+2:



x42 4x4 Matrix Mixer with input 1+2 routed to output 3+4:



x42 4x4 Matrix Mixer with input 3+4 routed to output 1+2:



How To - 7 Equal Latency for Tracks and Buses

[How To - Contents](#)

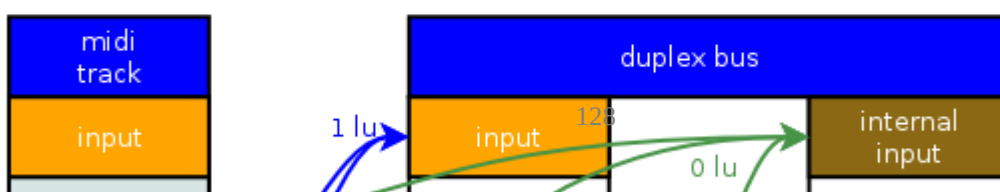
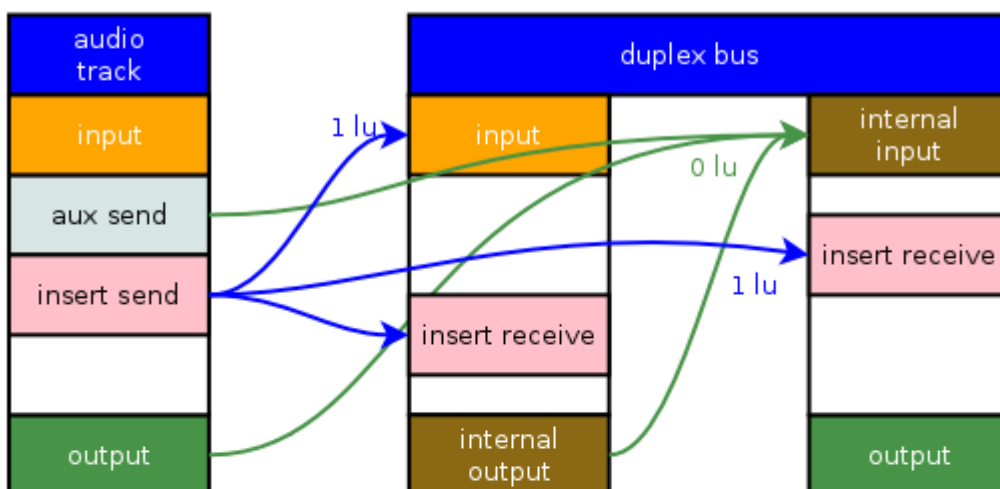
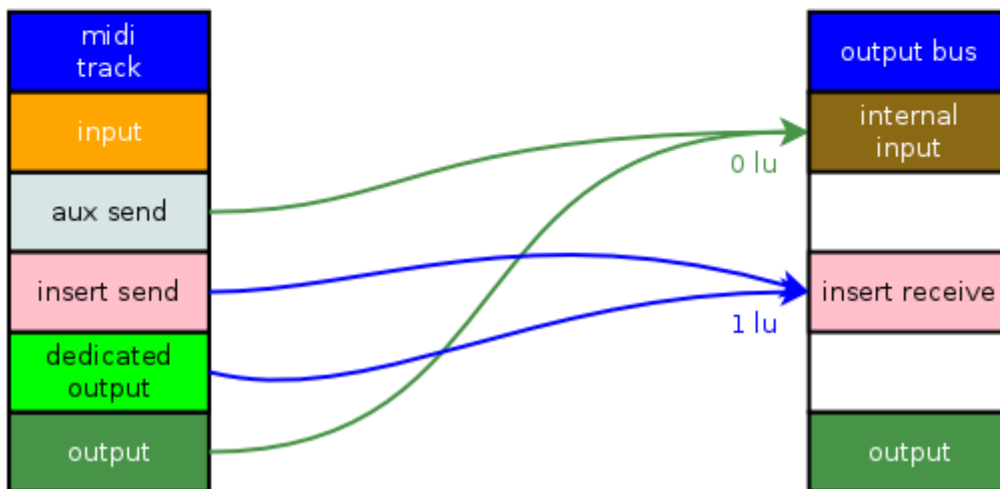
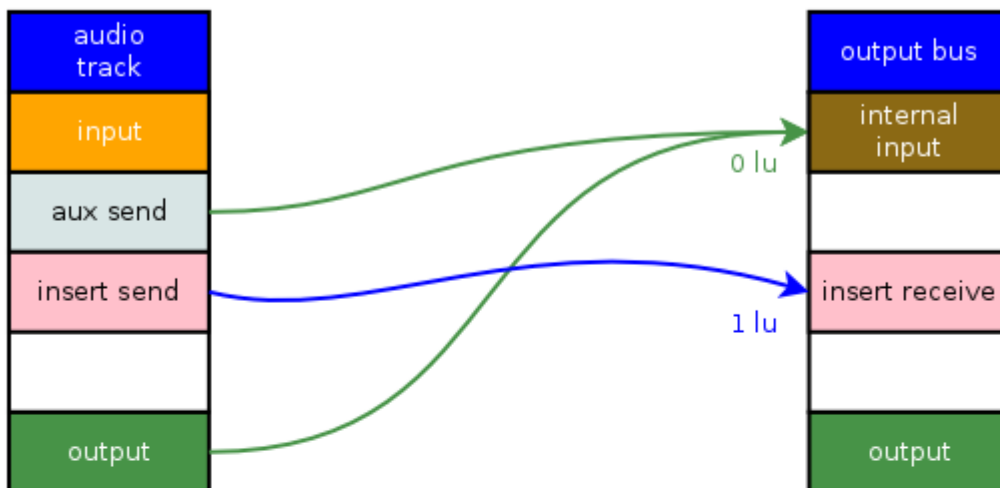
Note: With Qtractor 1.5.7 this How To has lost its importance because Qtractor's AUX Sends became much more flexible. Insert Sends are neither needed nor recommended for internal connections.

Equal Latency for Tracks and Buses

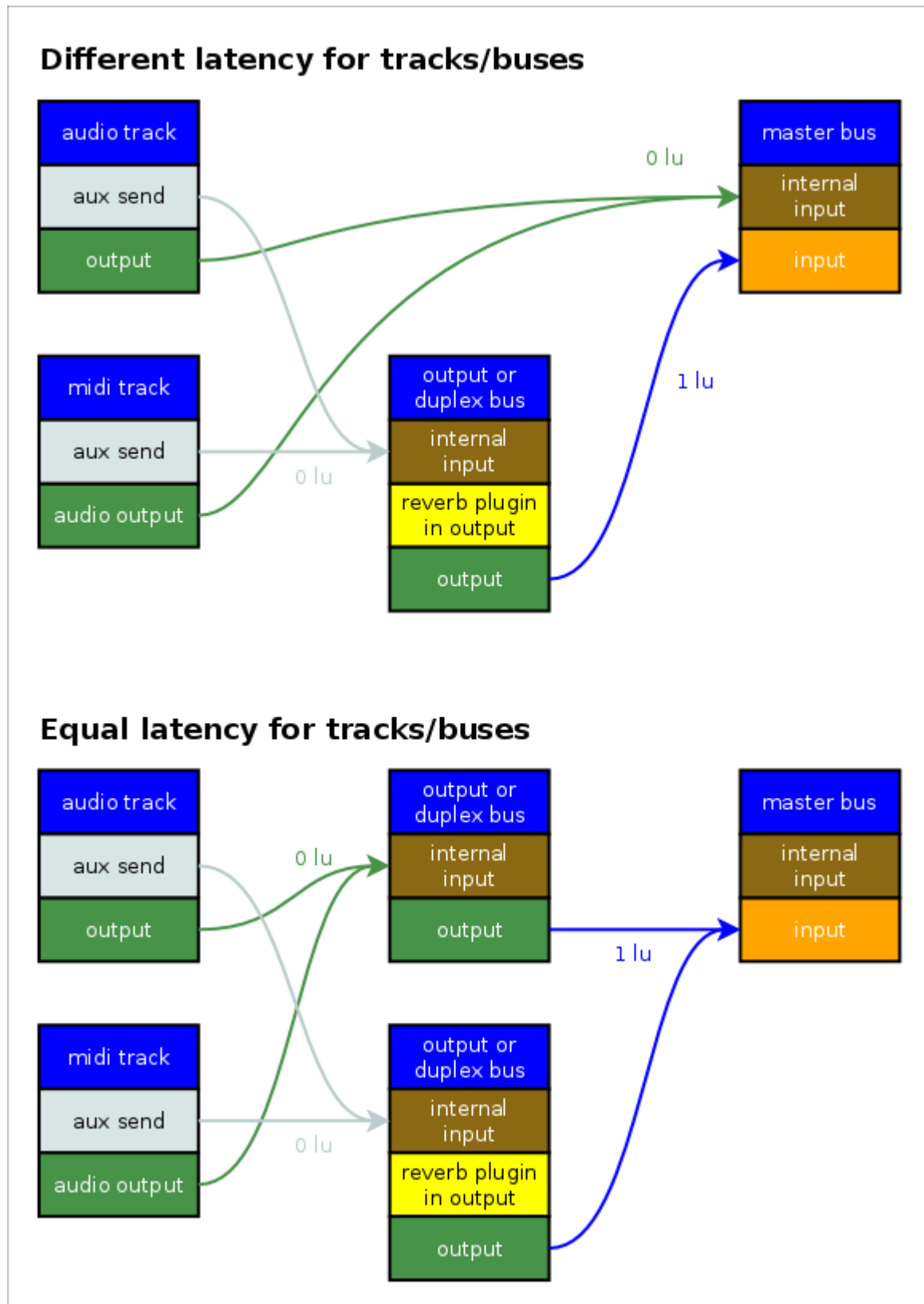
How long does it take for the audio material to get from the track to the output? Well, that depends. As a rule of thumb, there is no delay if you connect a track internally to a bus or use an aux send to a bus. Internal means that this connection is not visible outside of Qtractor, e.g. in Qjackctl.

On the other hand, there are connections that are visible from the outside, e.g. from the output of one bus to the input of another bus or with insert sends/receives. These connections have a latency that depends on the buffer size and the sampling rate used. If you operate jackd with a sampling rate of 48000 Hz and a buffer size (-p) of 1024, the latency is $1024/48000$ seconds = 21.3 ms. This is particularly audible with percussive sounds. In this How To we'll call it Latency Unit (lu), not to be confused with [Loudness Units as in EBU R 128](#).

Latencies with Qtractor



To get the timing tight and have everything in sync it's advisable to have the same latency for all tracks and buses. This simple example shows why you should not connect tracks directly to the master bus if plugins such as reverb are used in a separate bus that is fed by the tracks via aux sends.



If you have special needs then you can chain buses to be able to insert signals with differing latencies. But this is beyond the scope of this little How To.

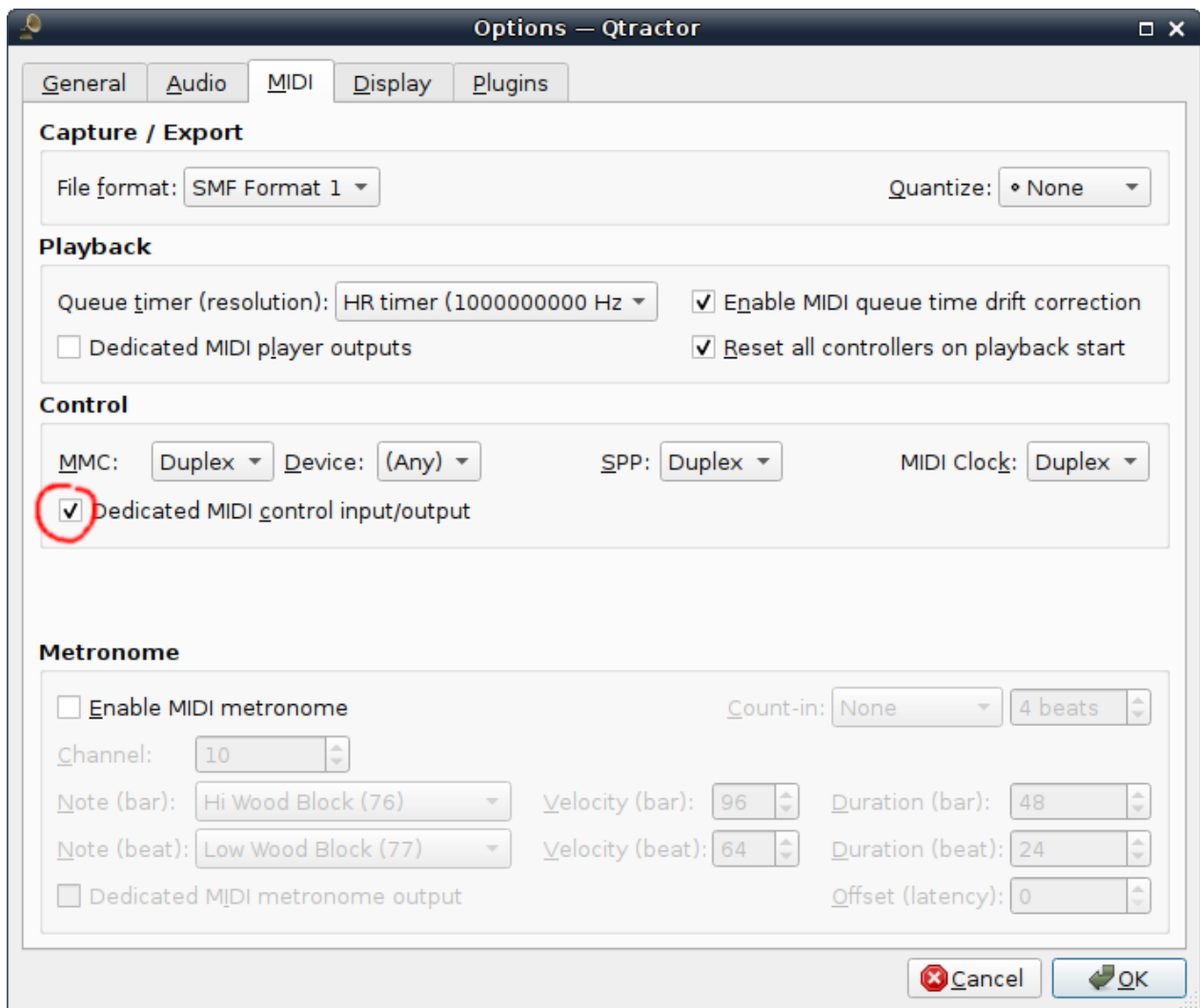
How To - 8 Controlling a Plugin's Parameter from a MIDI Clip

[How To - Contents](#)

Controlling a Plugin's Parameter from a MIDI Clip

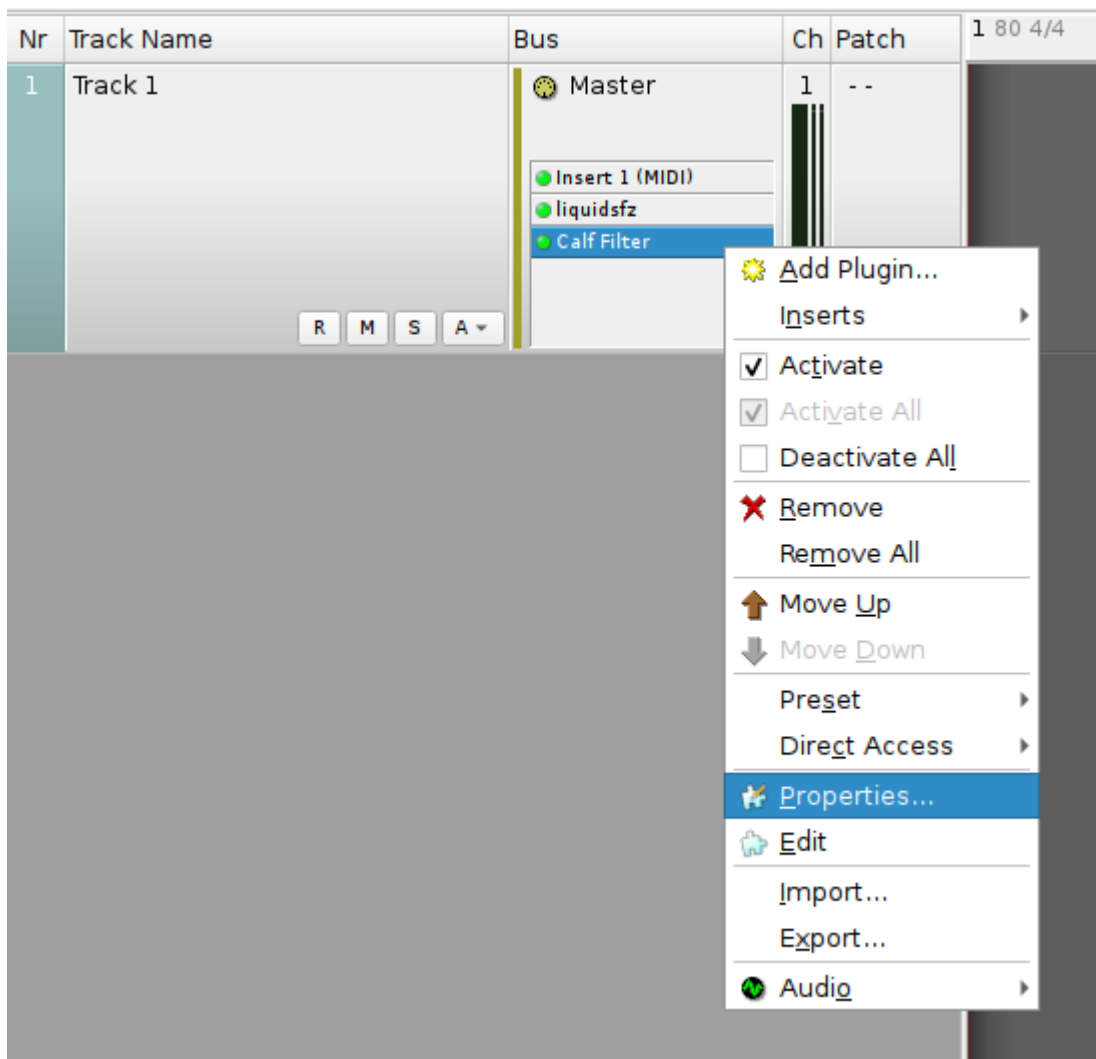
There are many ways to map a plugin's parameter to a MIDI CC. A simple one is using a MIDI Insert Send.

It's a good practice to have a dedicated MIDI control input in Qtractor. It reduces the danger of MIDI loops that can crash Qtractor. It can be activated in View -> Options -> MIDI -> Dedicated MIDI control input/output:

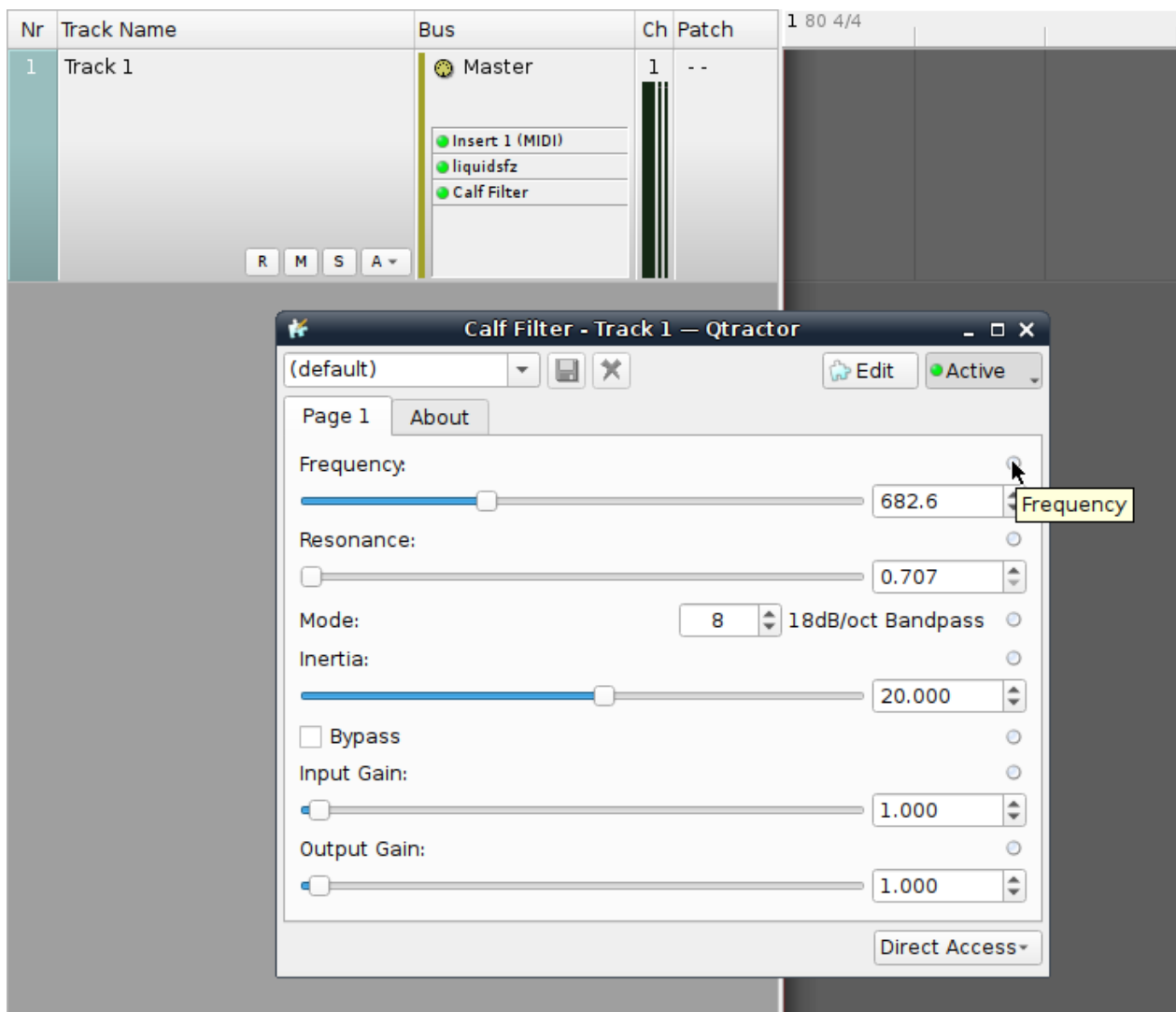


In this example we have a MIDI track with - a MIDI insert send - an instrument plugin (liquidsfz) - a filter plugin (Calf Filter)

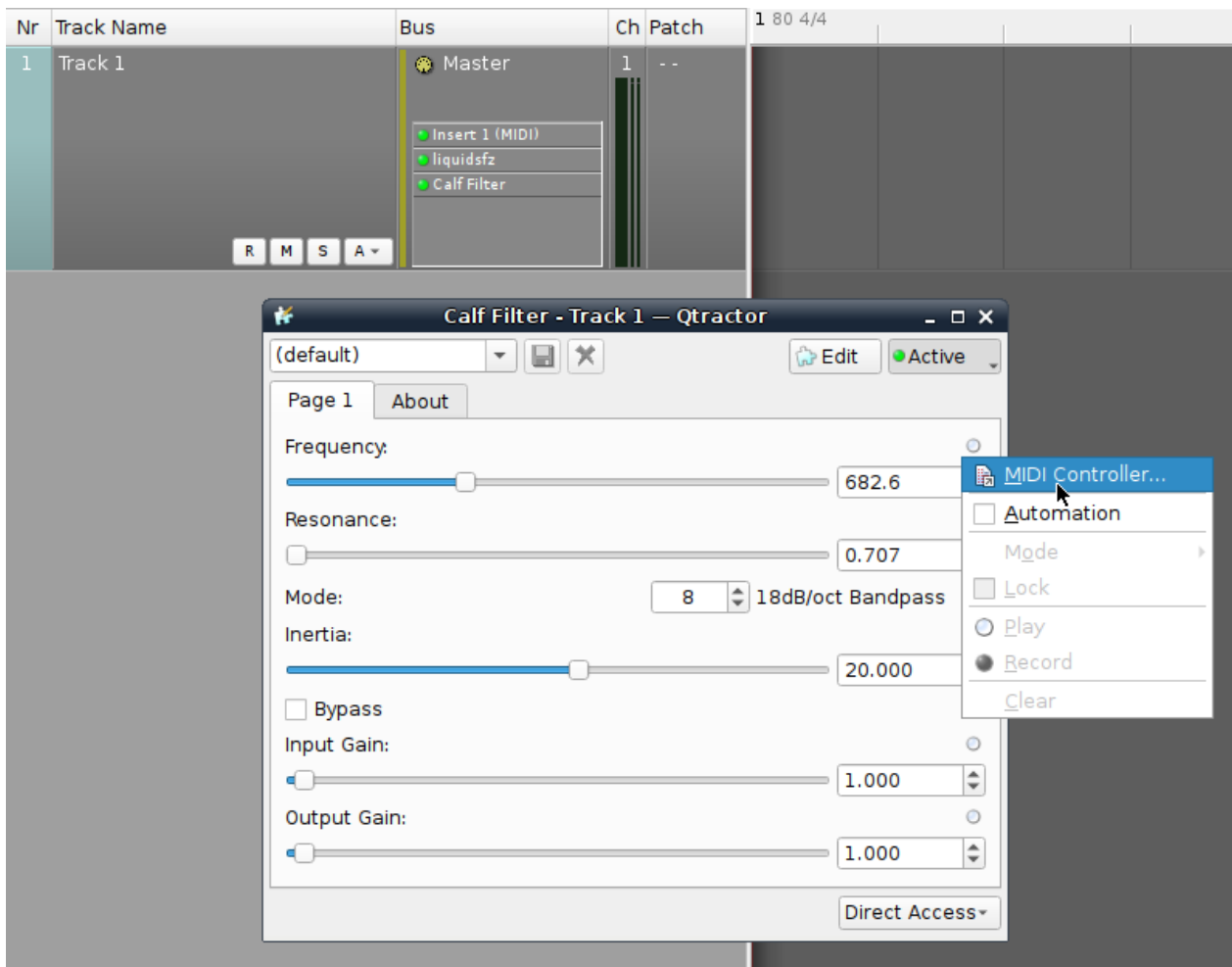
We want to control the filter plugin's frequency parameter with the modulation wheel. Right click on Calf Filter in the plugin box and select Properties...



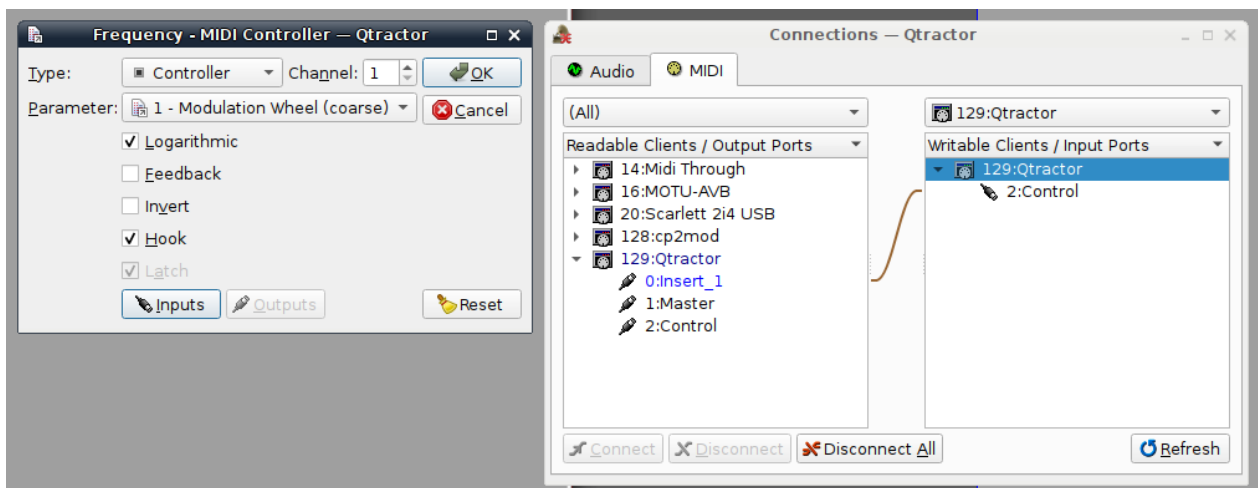
Click the clear LED of the Frequency parameter.



Click MIDI Controller...



Select MIDI Controller 1 - Modulation Wheel (Coarse), Logarithmic and Hook, then push the Inputs button, click the MIDI insert send (0:insert_1 in this example) and connect it with Qtractor's Control input.



Now you can control Calf Filter's frequency with your modulation wheel.

See this [video](#).

You can place a MIDI filter plugin between the MIDI Insert Send and the instrument plugin if you don't want the instrument plugin receive the modulation wheel data. But that's beyond the scope of this little How To.

How To - 9 Distributing Plugins' Load to multiple CPU Cores

[How To - Contents](#)

Distributing Plugins' Load to multiple CPU Cores

The real-time CPU load of plugins can be a problem. It can lead to xruns or require a big buffer size – or both. One of the workarounds is to try to distribute their load across multiple CPU cores. This is not possible with jack1. But jack2 can - under certain conditions - use more than one CPU core:

- There has to be more than one jack client. Qtractor is a single client so we have to move some plugins to another plugin host. Bonus points if we use multiple plugin hosts or plugin hosts that act as multiple jack clients.
- Audio data of different clients needs to be independent from each other. There is no way to parallelize the workload of a plugin chain.

A good use case is a Qtractor session with several buses containing reverb plugins. Those plugins can be moved to other plugin hosts, e.g. Carla with its engine running in Multiple Clients mode or [Non Mixer XT](#). Each mixer strip in Non Mixer XT can act as an own jack client, can host multiple plugins and assigns the mixer strip's name to the associated jack ports. Very nice.

Let's assume we have a Qtractor session that is already set up as in [How To - 7 Equal Latency for Tracks and Buses](#).

- Each track's audio is routed to a bus that is routed to the Master bus.
- Reverb plugins are in their own buses that are routed to the Master bus.

To parallelize audio processing we have to

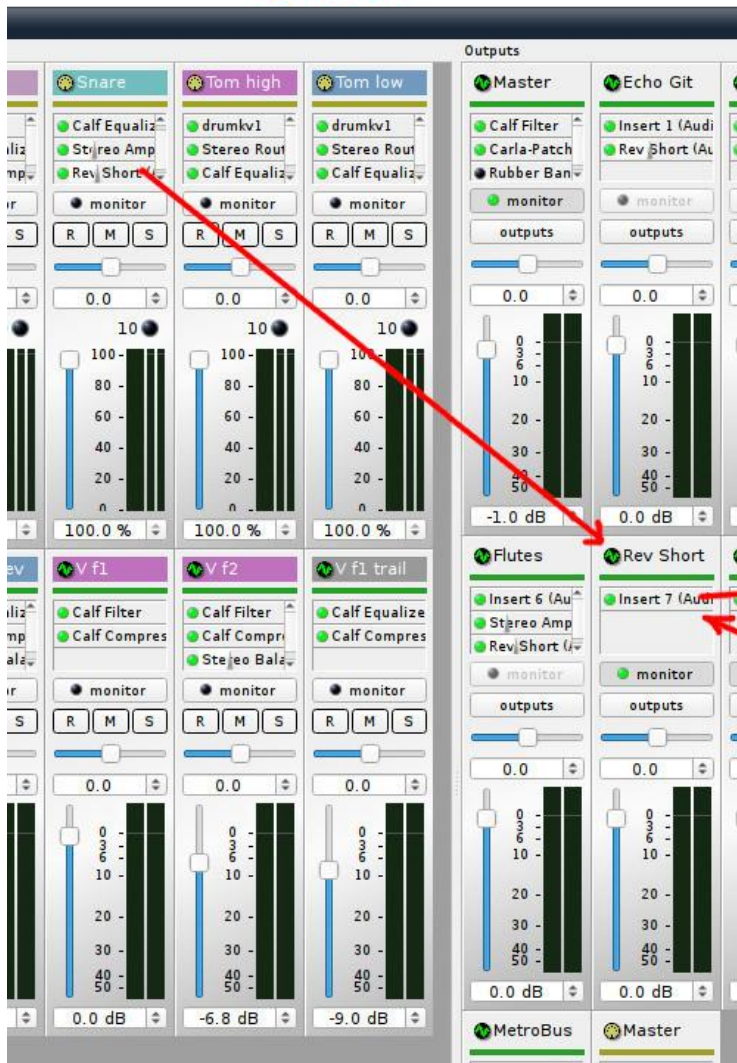
- move the reverb plugin (and maybe other plugins) from the reverb plugin's bus to a mixer strip in Non Mixer XT,
- put an Insert Send in the bus to and from the mixer strip. Dry Gain has to be set to 0.

An Insert Send adds a latency of one Latency Unit. So make sure that each bus has either an Insert Send/Return to a mixer strip or to itself (from an Insert Send to its own Return) to get the same latency for all buses.

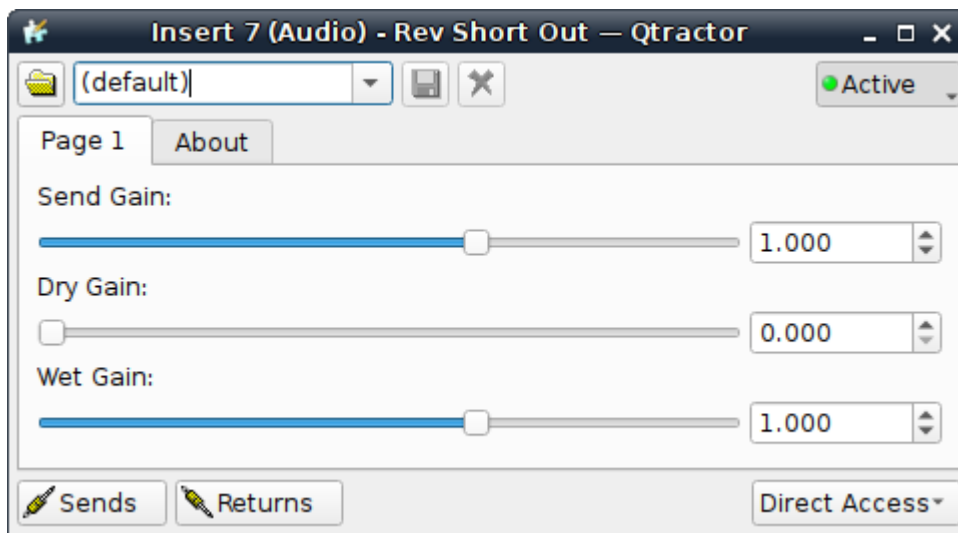
Moving plugins from tracks to Non Mixer XT should be possible as well. To get the same latency for all tracks you can use Insert Sends in all tracks or a bus with Insert Send (for tracks without an Insert Send) and a bus without an Insert Send (for tracks with an Insert Send).

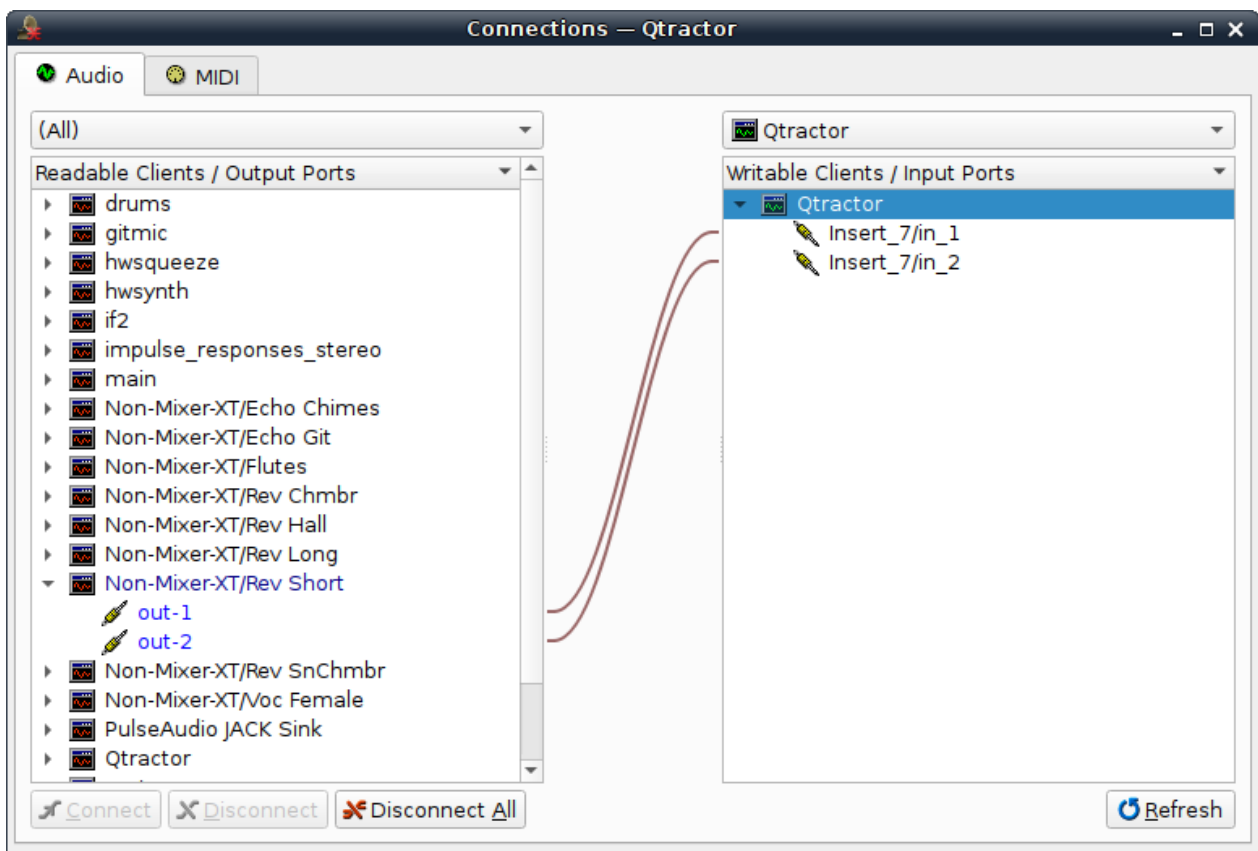
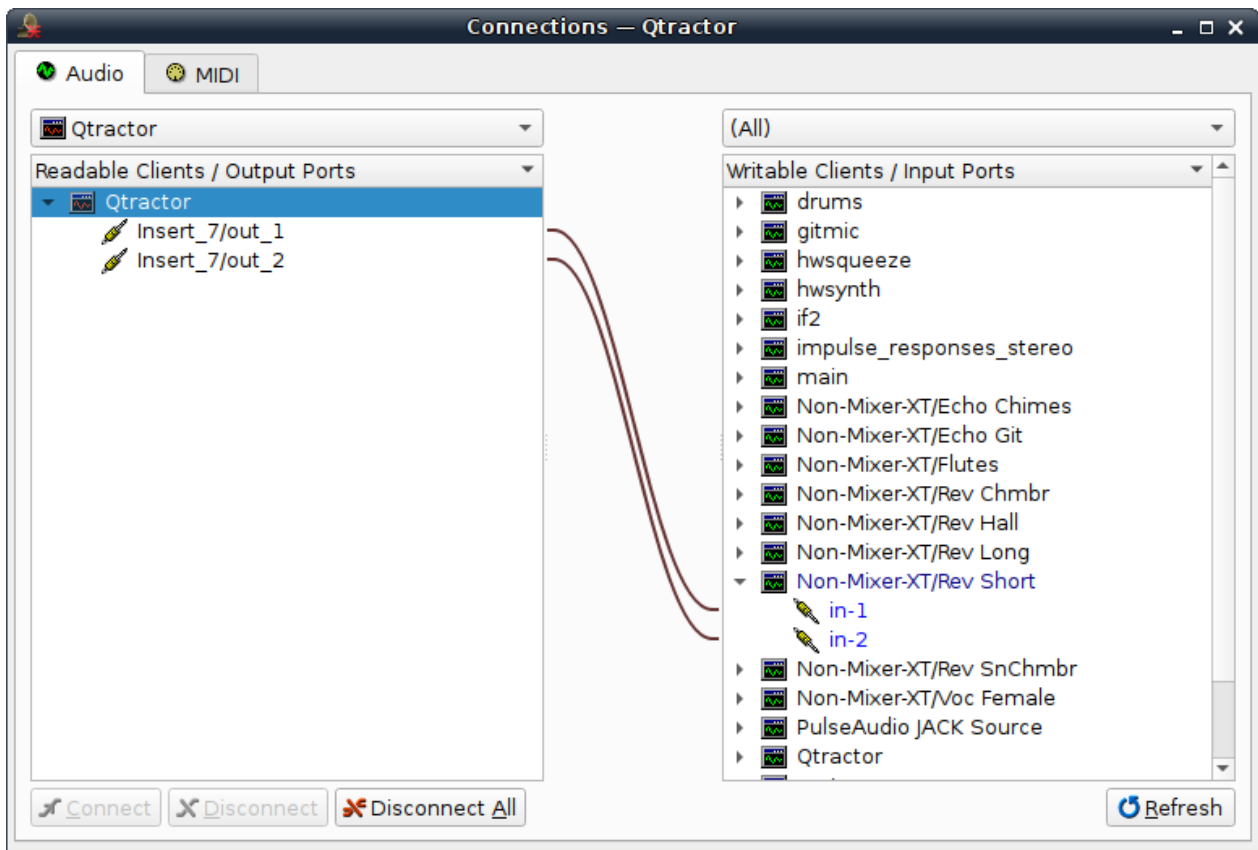
Start Non Mixer XT and open the corresponding project before you start Qtractor and load the saved session, otherwise the connections to and from the Insert Sends will not be established. Using a session manager to automate the setup is beyond the scope of the How-To.

Qtractor



Non Mixer XT





How To - 10 Get SOLO functionality on Buses

[How To - Contents](#)

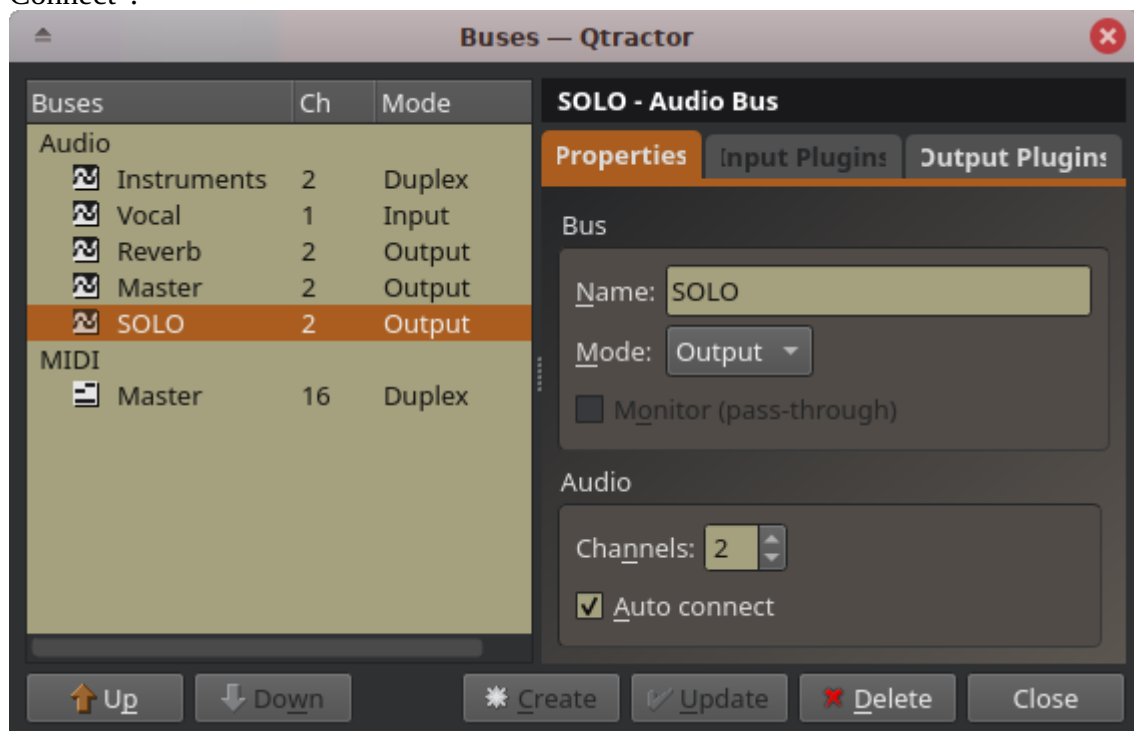
- **Author:** G3N-es.
- **Difficulty level:** Advanced.
- **It is assumed that you already know:**
 - [How To - 2 Create Individual MIDI and Audio Buses-Ports.](#)
 - Correctly arrange the buses to accept the Aux Send.

With this trick you will be able to listen in ‘SOLO’ mode:

- Tracks: Being able to isolate any point in the plugin chain.
- Buses: Being able to isolate any point in the plugin chain.

Configuration:

1. **SOLO Bus:** Create a new bus named “SOLO”, with “Mode: Output” and “Audio: Auto Connect”.



Important!!! Place it after your “Master” bus.

1. **Master Bus:** Disable “Audio: Auto Connect”.
2. **Aux Send to SOLO:** On the “Master” bus, at the end of your plugin chain, create an “Aux Send” to the “SOLO” bus. You should now be able to listen to your “Master” bus.

Usage:

Create “Aux Send to SOLO” to any point in your audio flow that you want to isolate for listening, whether it’s tracks or buses. If you place it at the beginning of a plugin chain, you’ll hear dry audio with no effects. If you place it at the end, you’ll hear all the effects (inserts and aux sends included). If you place it in the middle, you’ll be able to discriminate which plugins you want to hear and which you don’t. And all this without having to turn plugins on and off or use “SOLO - MUTE” buttons.

It’s that easy, that powerful.

Note: If what you need is the “MUTE” functionality on buses, the Carla plugin works exactly like a mute plugin in its default configuration.

How To - 11 Prevent Color Picker Freezing Qtractor

[How To - Contents](#)

Why does it happen?

If you use a GTK-based desktop environment, you will most likely encounter this annoying issue: When using color pickers in Qtractor, it freezes or crashes.

The reason: desktops based on the GTK framework can create conflicts and integration problems with applications based on the Qt framework.

Solution

There is an application specifically dedicated to these integration problems.

You can find it as “qt6ct” (Qt6 Configuration Tool) or “qt5ct” (Qt5 Configuration Tool), depending on the Qt version you want to integrate.

So to know which one you will need, you must know the Qt version that was used to compile your version of Qtractor.

Steps

1 Qt version of your Qtractor:

You can find it in the Qtractor help menu.

2 Install the Qt configuration tool:

Install “qt6ct”, or “qt5ct” as appropriate. Open your terminal.

In case of Qt6:

```
sudo apt-get install qt6ct
```

In case of Qt5:

```
sudo apt-get install qt5ct
```

3 Create the QT_QPA_PLATFORMTHEME environment variable:

With this variable we will tell the environment that the visual themes of the Qt applications should be managed by the Qt configuration tool that we just installed.

To create the variable we run the following in the terminal:

In case of Qt6:

```
export QT_QPA_PLATFORMTHEME=qt6ct
```

In case of Qt5:

```
export QT_QPA_PLATFORMTHEME=qt5ct
```

The issue has been resolved.

How To - 12 Automate Buses Experimentally with MIDI filters

[How To - Contents](#)

- **Author:** G3N-es.
- **Difficulty level:** Advanced.

NOTE!

Since Qtractor version 1.5.6, there are simpler ways to perform these tasks. The procedure in this tutorial is deprecated. However, it shows alternative methods and workflows with MIDI filters. Therefore, for educational and experimental purposes, it still makes sense to retain it.

The recommended method for this task can be found at: [How To - 14 Automate Buses and Tracks by Layer with Insert Controller](#)

Ingredients:

- Qtractor 1.5 or higher.
- [Plugin Host: Carla-lv2](#)
- [Plugins MIDI filter](#)
- Template: [MIDI-ControllerSurface.qtt](#)

Introduction

In Qtractor the default way of automating is by tracks. Each track allows you to view only one automation at a time. However sometimes we need to automate buses, or view several layers (tracks) of automation at the same time. Is this possible? Yes, of course.

Qtractor offers modular functionalities. If we relate them together creatively we can achieve almost anything we need.

In this case the modules are:

- MIDI tracks that send standardized CC events from their faders.
- Automation of faders in tracks.
- MIDI controller (Learn MIDI): Allows you to control any plugin via CC, even if the plugin does not have MIDI command reception implemented in its controls. It also allows you to control any element of Qtractor itself.
- Complete and flexible MIDI signal routing system.
- Supports MIDI filtering plugins.

The basic concept for automating buses is:

1. Link a fader of a MIDI track to the element you want to control.
2. Automate the fader. The linked element is therefore also automated.
3. This allows us to create independent layers (tracks) for each automation.

We encounter two obstacles:

1. The MIDI faders emit only 2 standardized events (Volume:CC7, Pan:CC10) and we need to send any MIDI event.
2. The midi track sends and receives the same CC message on the same MIDI channel. Activating the “Control” port generates an infinite loop that will make Qtractor crash.

These obstacles are solved with “midi filtering plugins”, and a “host plugin” that prevents midi signal leaks.

Template MIDI-ControllerSurface.qtt

Filtering and interconnecting MIDI signals is a very broad, complex and technical thing. It takes several books to explain it. Therefore, this tutorial has the following focus:

1. We offer you a configured and fully functional template.
2. We explain how to use it, edit it and create your own versions of it.
3. In this way, you will expand your knowledge by creating your own configurations.

This is more manageable than having to explain the why and how of each connection and filter. This template allows you to have an “Automatable Midi Controller Surface” within Qtractor. This is the power of modularity in Qtractor. It allows us to create workflows that work almost like a native functionality developed in C++.

INSTRUCTIONS MIDI-ContollerSurface.qtt

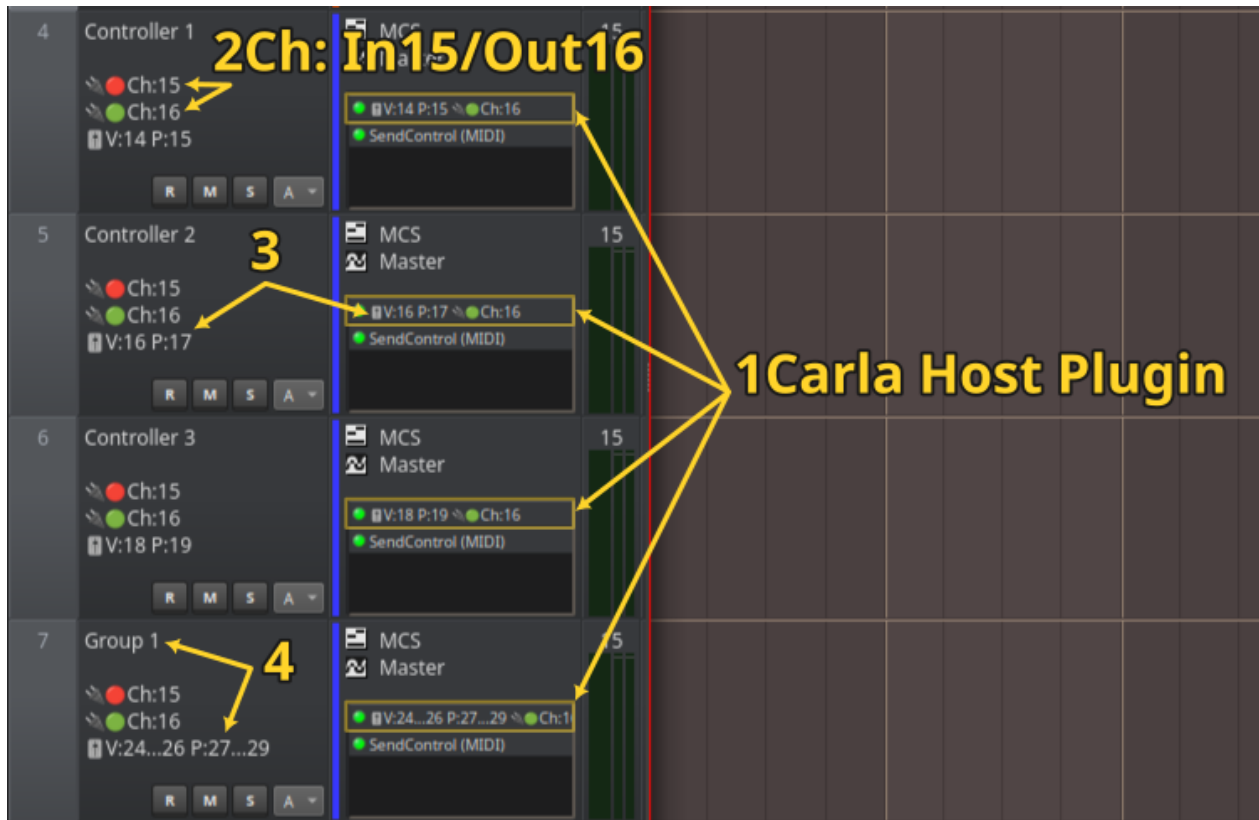
In Qtractor activate: Options > MIDI > Control > “Dedicated input/output for midi control”. Open “MIDI-ContollerSurface.qtt”.

PRECAUTIONS!

To avoid Qtractor crashing due to MIDI event leaks we must take the following precautions:

- Never deactivate the first plugin (Host Carla) of Controller/Group tracks.
- If we need more MIDI plugins on Controller/Group tracks they must be hosted in Carla, never directly in the plugin box.
- MIDI channels 15 and 16 are reserved only for control/automation tasks. Do not use these channels on generic midi tracks.

Template Preset Configuration

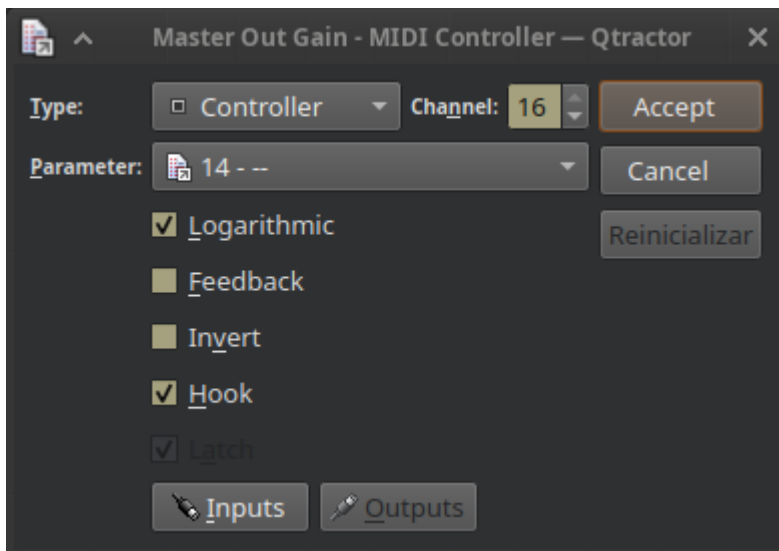


1. Carla is the first Controller/Group tracks plugin. It performs MIDI channel filtering and replaces CC events via the plugins it hosts. As you can see, it does not display its name, but rather the functionality it performs. This is achieved by assigning an alias in the plugin properties.
2. All Controller/Group tracks receive events on the original MIDI channel 15. However, thanks to filtering, they output on channel 16. This prevents possible infinite loops.
3. Both Carla and the Controller/Group tracks indicate which CC messages they are outputting. The image indicates V:16, P:17. This means:
 - V:16 = Volume Fader outputs CC16
 - P:17 = Pan Fader outputs CC17
4. Group faders output multiple CC messages from each fader. This allows groups to be created:
 - V:24...26 = Volume Fader outputs CC24, CC25, CC26
 - P:27...29 = Pan Fader outputs CC27, CC28, CC29

Practical Example: Assign the reception of a CC event

Let's imagine that we want to automate the Master audio bus Gain fader. The "Controller 1" track indicates that its Gain fader outputs CC14 on channel 16.

Right-click on the Master audio bus Gain fader:



1. Assign Type Controller, Channel 16 and Parameter 14.
2. Logarithmic and Hook must be checked.
3. Move the “Controller 1” Volume fader: We see that they are already linked.

If we want, we can automate the “Controller 1” Volume fader, and therefore, since they are linked, the Master bus Gain fader. To automate tracks, consult the Qtractor manual: [4.6.8. Automation](#)

Correct settings by destination controller type:

- **Audio Gain:**
 - **Logarithmic:** Checked
 - **Hook:** Checked
- **Midi Volume, Audio/Midi Pan:**
 - **Hook:** Checked
- **Plugins:**
 - **Logarithmic** (Try in each case)
 - **Hook:** Checked

Trick Crossfader: To perform crossfader, control several elements from a Group and checked the option “Invert” on the one you want to crossfade.

Customization

To add Controller/Group tracks, duplicate the existing ones and modify the CC number into Carla in edit parameters.

Recommended Assignable CC:

To avoid conflicts we recommend using only “general purpose controllers” and “non-predefined controllers”. This would be the available list: 3,9,14...31,41,46...63,85...90,102...119

Remember to also edit the plugins name, the Carla’s alias and the track name with the new CC number to always have the information clear at sight and avoid confusion.

To modify the existing Controller/Group tracks, act in an analogous way.

It may seem cumbersome, but the goal should be to create custom presets, and once you've set them up you don't have to touch them again.

Once you've got everything just right, you can save each track configuration as a preset. Track presets are saved by right-clicking on the "plugin box" > Export. And they are loaded by right-clicking on the "plugin box" > Import. You can also save a preset for the entire session by saving it as a template.

Other Uses

You can also try to perform different types of filtering for different functionalities.

For example:

You can create a configuration to control devices external to Qtractor (synthesizers, lighting devices, motors, etc). In these cases channel filtering is not necessary, because there is no risk of looping.

The potential is endless.

How To - 13 Make Successful Audio Connections

[How To - Contents](#)

- **Author:** G3N-es
- **Resources cited in the tutorial:**
 - SEND/RETURNS plugins from the LSP suite: <https://lsp-plug.in/?page=download>
 - Cadence-Render Recorder: <https://kx.studio/Applications:Cadence-Render>

Qtractor, although it performs the same task as other DAWs and shares the same nomenclature, has its own unique approach.

This can lead to confusion, as we try to apply pre-learned methods from other DAWs that don't fit with Qtractor. Ninety percent of the time that I thought there was a error in Qtractor, there wasn't. On the contrary. I simply didn't understand its logic. Once I understood it, I discovered that the solution proposed by Qtractor was more fluid and efficient.

In this tutorial, I'll try to give you an overview of how connections work in Qtractor so you can develop your own audio flows with complete freedom.

Connection Areas

When we talk about connections, Qtractor distinguishes between two defined areas.

1. Recording and playback connection ports.
2. Mixing connections.

1 Recording and Playback Connection Ports

Input and output ports on buses are not intended for mixing. As much as this statement may surprise you, read this section. You'll understand.

In other DAWs, the way to connect to a recording and playback device is through a form called "Devices" or similar. In these DAWs, the concept of a port may not even appear. In Qtractor, no such form exists. The connection ports to external devices are the input and output buses. Connections are managed from the "Connections" window. This causes a lot of confusion, especially for people coming from other operating systems. The first thing they do is look for the "Devices" form, and of course, they can't find it.

However, the approach in Qtractor is much more powerful. You can create as many input and output ports as you want for different purposes. For example, you can create an input bus:

- Mono for vocals with its plugin chain ready to record the already processed vocals.
- Another stereo for guitars with its own effects chain.
- Another stereo for general purposes.
- Etc., etc., etc.

You open it and start recording with everything already configured. I don't know if other DAWs offer this feature. I know there are DAWs that can do it by track, but not by **groups**.

Groups. Here's another difference: In Qtractor, tracks are hosted by buses (groups). There can't be independent tracks. In fact, **input buses** are all recording groups. The "default" bus in Qtractor (pre-named "Master") is actually a group that hosts new tracks by default and assigns inputs and outputs ports to them when they are created. **Output buses** can serve as group buses, auxiliary buses, or master buses depending on how they are used. They don't even have to be configured in any specific way. They are self-defined by their use. Everything is defined because it has its own sense.

It's that simple and logical:

- If they host tracks, they serve as "Group."
- If they don't host tracks and receive parallel signals, they serve as "Auxiliary (Effects) Buses".
- If they don't host tracks and concentrate mix signals, they serve as "(Auxiliary) Mix Buses".
- If it's the bus that connects to the end of the audio chain, and is connected to an external device intended for playback, it's the "Master Bus".
- A single bus can perform several of these functions at the same time.

Once users figure this out, they run into another reasonable confusion: **"If I create a bus, and it's duplex and has an input and output... The input bus is the input of the duplex bus where I should connect the signals to be mixed."**

In addition to recording, they'll try to use the input bus to mix signals... No, it doesn't work that way. The input and output ports in Qtractor aren't designed for mixing, only for connecting to external recording and playback devices, whether hardware (a USB microphone) or software (a softsynth).

You might be asking: "But aren't output buses designed for mixing either? I don't understand." My answer: The **plugin rack** in output buses is designed for mixing. The output ports (including the faders) are not.

"But then what's the point of buses having faders?" For example, for the Master bus, which is the one that connects to the outside. And for the others too, but for connecting to devices, not for mixing.

Imagine: You want to collaborate with a friend because he's good at mixing and mastering. However, he works with a different DAW (we all know your friend has his quirks, but we love him just the same). You'll need to freeze/render your groups and aux buses. That is, you'll need to record your buses to an external device on separate tracks so they can be seamlessly imported into in your friend's DAW. You might find the faders useful in this process for adjusting volumes and panning.

Other cases: Connecting specific buses to external devices for monitoring purposes. Etc., etc.

They've never lost their purpose. However, they're not necessary for the mixing process.

"But do I need faders on the buses to mix!!!" Add gain and panning plugins. Those are your faders.

"Are you telling me I can't use the bus outputs for mixing?" I'm not saying you can't. I'm saying they weren't designed for that purpose.

“What if I try to record the master output to a track to get my master copy?”

WARNING!!! Do not use **Tracks / Export Tracks / Audio** for this. As its name suggests, it is only valid for tracks (exporting the direct signal of the track through its Output Bus/Group). It ignores Aux Sends between buses. It will not work for export buses with auxiliary and/or mixing function.

Wow! I do it that way too, recording in the session itself. Indeed, the best way to create a copy (a render) of the “Final Master” is to record the session in real time. This applies to any DAW. It’s the only way to ensure that exactly what you hear is recorded...

But... it won’t work. It’s not an external device: As we’ve already said, these ports are designed to connect to external devices.

However, **you can record your final Master in Qtractor if :**

- Create a external audio input and output ports, that bridge the Master Bus and a dedicated Export Bus. In “Summary | Conflicting Connections and Troubleshooting” we explain several ways to create these external ports.
- Use third-party plugins: Send (for Master Bus) and return (for Export Bus) . The SEND and RETURN plugins in the LSP suite do a great job. <https://lsp-plug.in/?page=download>
- If your engine is PipeWire: Record the audio from the “system monitor” to your “Export Bus”. You just need to make sure your Qtractor session is the only one outputting audio to the system at that time.

Others prefer to use external recorders for this task: SoundRecorder, Cadence-Render, etc.

Although, **recording in the session itself has advantages:**

- Precisely define the start and end of the recording with the Punch function in the Qtractor transport.
- The file is automatically saved in the same project folder.

We’ll cover the mixing area in the next section.

2 Mixing Connections

The mixing area in Qtractor can be said to covers the tracks (including racks and faders) and the plugin racks for the buses.

Qtractor provides us with the following mixing tools:

Direct Connection to the Bus

Logically, each track’s output is connected to the bus that hosts it. Additionally, **tracks and buses MIDI** can also be joined to audio groups, as they can host softsynths with audio outputs. In this case, the output bus is assigned with right-click on the plugin rack: Audio > Bus to Select.

Aux Sends

Aux Sends allow you to send signals from track to bus and from bus to bus. They are pseudo-plugins (internally predefined plugins). They are accessed by right-clicking on the plugin racks: Inserts > Audio > Add Aux Sends.

They have one limitation: They don't allow connections between tracks. For sidechain connections and to simplify connections, it's sometimes useful to interconnect tracks. Qtractor has never had a tool for this purpose.

In those cases, direct connections between pseudo-plugins **Inserts** (Inserts > Audio > Add Insert) could be used, which we'll discuss below. However, this was never their intended purpose for **Inserts**, and it was always considered inadvisable to use them this way. Despite everything, since there was no other way, we used them. In fact, there are even "How to" guides on this wiki that describe their use to interconnect tracks.

Interconnect **Inserts** directly may work in Pipewire, but not in Jack2 or vice versa. Or maybe one, or the other, (or even Qtractor itself) has an update and stops working. They weren't designed for that job. It's that simple. This can mean that old sessions stop working and need to be rewired.

Fortunately, there are alternatives for this task today. Once again, the SEND and RETURN plugins in the LSP suite. In addition to simplifying these workflows, they work directly with other plugins in the same suite without the need to set up dedicated connections. Using "Inserts" to interconnect tracks no longer makes sense.

Inserts

If "aux sends" are used to make dedicated internal connections to the mix, **"inserts" are used** to integrate external devices in the mix. That is, you can use external software devices (effects pedals like Rakarrack, etc.) and hardware devices (a compressor, etc.) in real time as if they were internal plugins.

If you directly connect one insert to another, you will create an internal connection, not an external one. Therefore, there's no guarantee that it will work correctly.

Summary | Conflicting Connections and Troubleshooting

You guessed right. Anything that involves connecting inserts to each other, buses to each other, or buses and inserts to each other may not work. This is not recommended. These elements were never designed to be interconnected, but rather to be connected to an external device.

This is precisely the solution. Specific situations may always arise where make these connections are useful: record parts of tracks on other tracks, obtain a copy of the final master, etc. Place an external device (input and output ports) between them. Don't connect them directly.

You may be wondering: **"How do I create an external device with audio inputs and outputs that allows me to create the bridge?"**

- The Carla application has its own "Gain Plugin"; load it and make the connections through it.
- If your engine is PiPeWire, it's simpler and more versatile. You can use the ALSA command:

```
pactl load-module module-null-sink sink_name=BRIDGE  
sink_properties=device.description="BRIDGE"
```

This will create a virtual device with the external ports. The "sink_name" sets the semantic identifier of the device, and "sink_properties" sets the name visible in connections. Another advantage is that you can add this command to your linux "Automatic Application Startup

Session”, so it will appear every time you login your computer. This allows you to save the interconnections in your Qtractor template.

Freedom above all.

How To - 14 Automate Buses and Tracks by Layer with Insert Controller

[How To - Contents](#)

• **Author:** G3N-es.

Introduction

Since Qtractor version 1.5.6, has been included the pseudo-plugin: Insert > MIDI > Controller. This allows you to generate and send automatable MIDI control signals from tracks. This means that Qtractor can act as a “Virtual Automatable MIDI Control Mixer”. This applies to both external devices (audio creation software or hardware, lighting systems, robotics, etc.) and internal Qtractor buses and tracks (including plugins, of course).

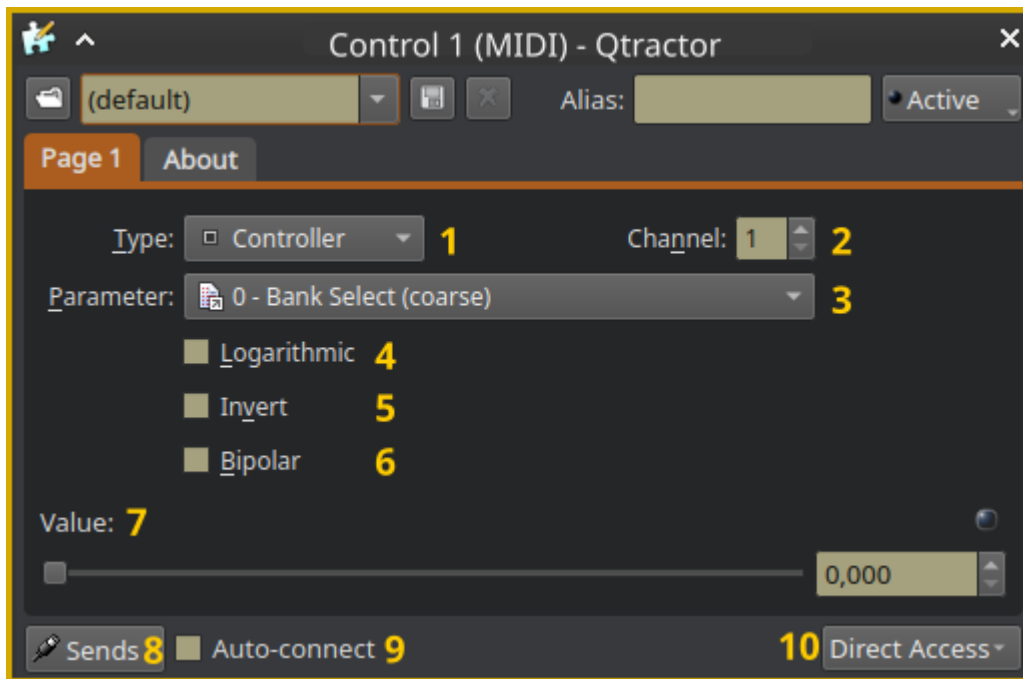
The possibilities are now endless: imagine, experiment, create, and have fun.

In this tutorial, we’ll show you the Controllers and how to configure them to automate buses and tracks by layer. You can have each automation element in a dedicated track. This makes it easier to view and edit automation. You can also automate buses in a user-friendly way, following the existing workflow in Qtractor.

Controllers allow you to:

1. Create individual tracks for each automation.
2. Automate Tracks.
3. Automate Buses.

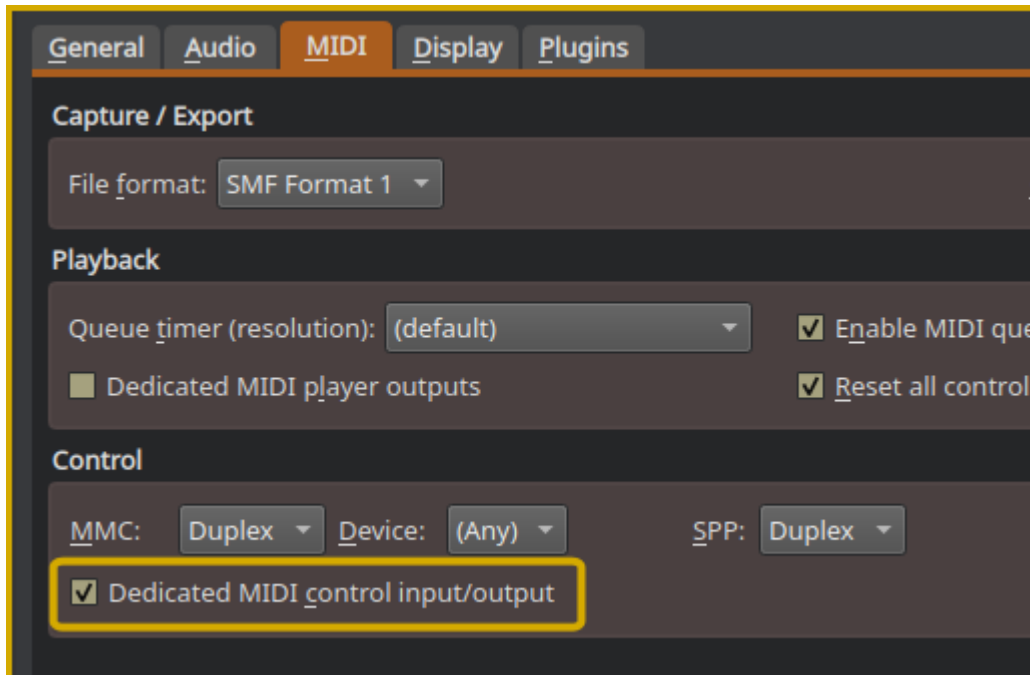
Controller Properties



1. **Type:** I recommend "Control 14". Its higher resolution allows for optimal adaptation to any target value.
2. **Channel:** MIDI send/receive channel.
3. **Parameter:** It's more organized to use them according to their pre-assigned functions, but it's not mandatory.
4. **Logarithmic:** Modifier suitable for controlling audio gain.
5. **Invert:** Modifier useful for crossfade and other effects.
6. **Bipolar:** Modifier for controlling pans/balances.
7. **Value:** Value to be sent.
8. **Sends:** Opens the connections window for sending control messages (internal and external).
9. **Auto-connect:** Connects the **Controller** to Qtractor's control message input.
10. **Direct Access:** Allows you to control the send value directly from the plugin rack.

Settings:

Qtractor



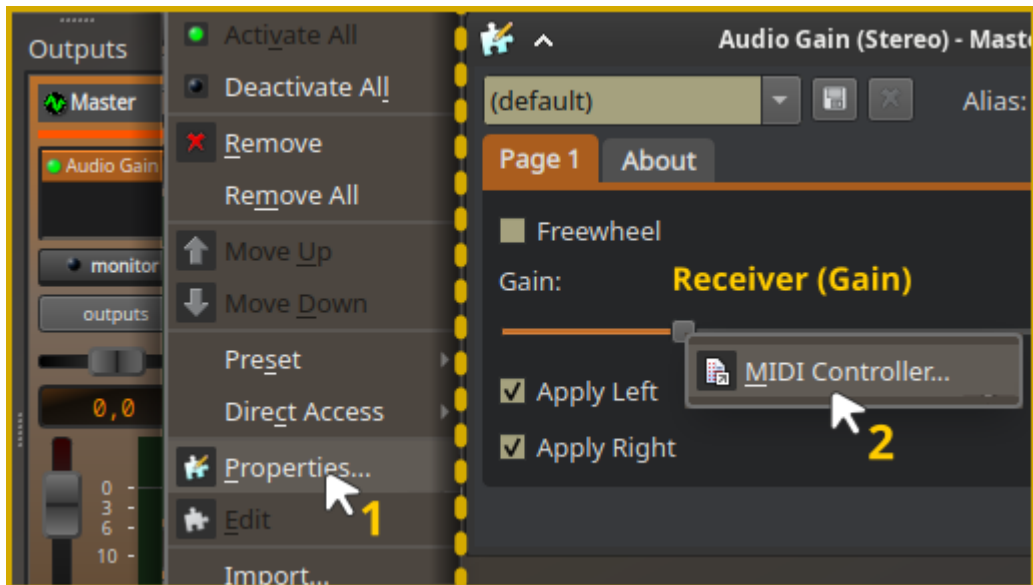
In View > Options > MIDI { Control, check the “Dedicated MIDI Control input/output” box.

Qtractor now has a dedicated, reliable input port exclusively for receiving MIDI controls. There, we’ll connect our **Controller** pseudo-plugins using its “**Auto-connect**” property.

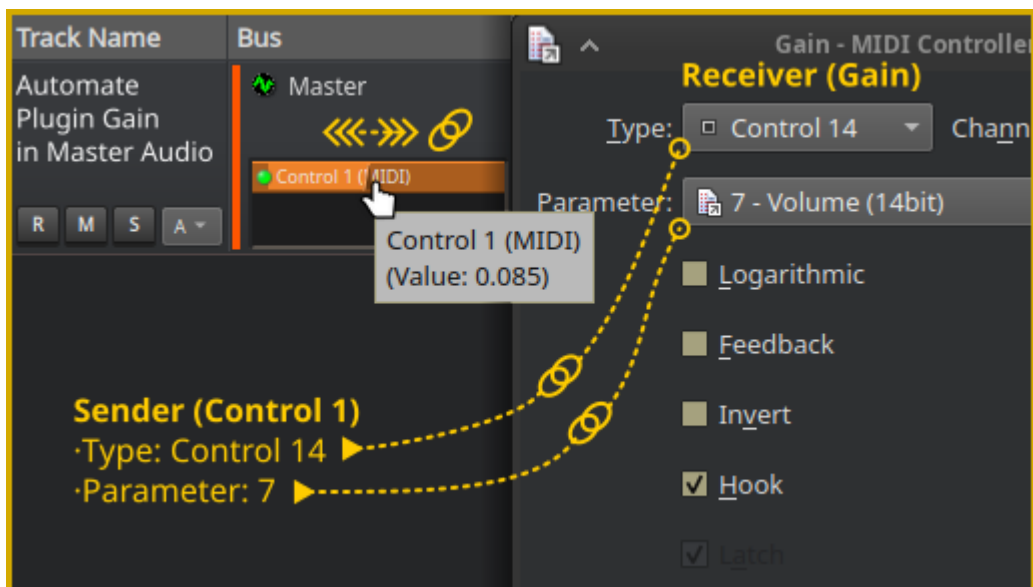
Sender (Controller)

1. **Create a new track:** It can be MIDI or audio. Identify it with the function it will perform.
Example Name: “Automate Plugin Gain in Master Audio”.
2. **Add Controller:** Right-click on the plugin rack (Inserts > MIDI > Add Controller).
Activate it.
3. **Configure the Controller properties:** Check the “**Auto-connect**” box. Choose the type, parameter and modifiers that best suits the receiver.
4. **Direct Access:** Create a shortcut to “**Value**” so that it is accessible from the plugin rack.

Receiver (Property of the plugin to automate)

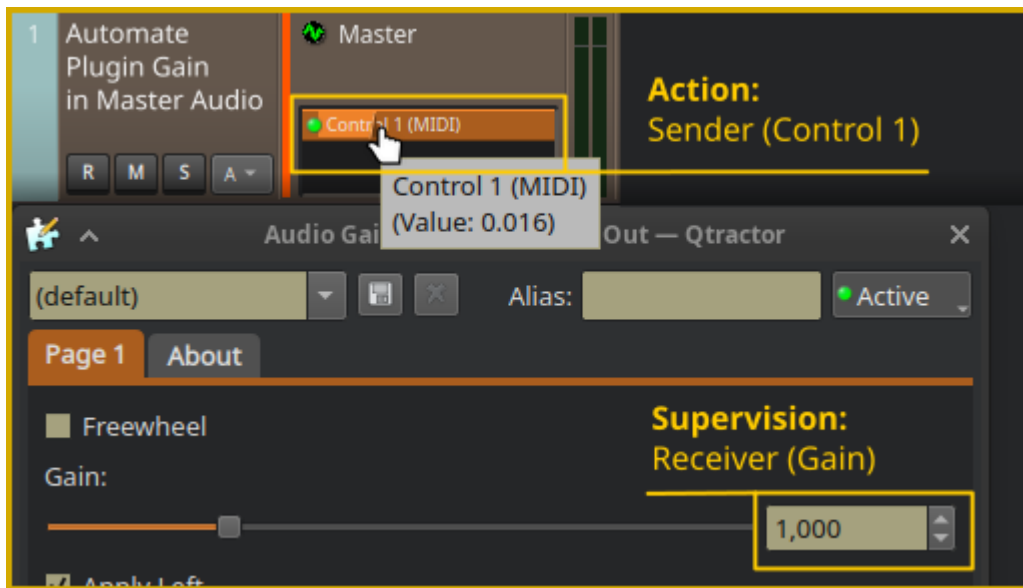


1. Go to the properties of the plugin or element you want to automate.
2. Right-click on the property you want to automate and click **“MIDI Controller”**.



Move the slider on your sending Controller. The receiver will automatically be preconfigured. This is known as “Learn MIDI” in other applications. Click OK to set the link configuration. If you move the sender, the receiver will reflect each action.

Calibrate



In the image, we can see that the Controller value: 0.016, is equivalent to the default Gain value: 1.000. Now, the Controller is calibrated. When we start the automation, it will start with the plugin's default Gain value.

Each plugin has its own range of values: maximum, minimum, and default. This is up to each developer. We can find gain plugins that allow a 20 dB boost, others only 6 dB. Others don't specify values in dB, but rather in scales (as in this case).

That's why you'll need to calibrate the send. It's as simple as moving the send slider until the receiving slider is set to its default value. From there, you'll make the changes on the send slider, but you'll watch (monitor) the changes on the destination slider.

Simply:

1. Make the changes from the Controller's "Value" shortcut on your mixer.
2. Keep the plugin GUI open to see the changes you're making.

It's easier and more intuitive to do than to explain :).

Automate

Everything is now set up and calibrated. All that's left is to automate the Controller's "Value" from the track, just like any other plugin. If you don't know how, consult the Manual [4.6.8. Automation](#).

Final Remarks

Imagine, experiment, create, and have fun.

How To - 15 Use Multiple SoftSynths on a Track

[How To - Contents](#)

- **Author:** G3N-es.
- **Difficulty level:** Easy.

Problem

Mixing multiple softsynths plugins is a intuitive way to customize timbres. However, if you try to load multiple softsynths on a track, most likely only the last one will play. This is because softsynths generally don't have audio inputs, only outputs. The second softsynth cuts the audio signal from the first. There are exceptions, such as the **“Vee One Suite,”** which allows you to stack multiple synths on a single track.

Solution

If you encounter this problem, there is a easy solution. Toggle softsynths with **“Aux Sends”** to the audio output bus of the MIDI track.

A simple example in the illustration:



How To - 16 Obtain a Multi Ghost Reference

[How To - Contents](#)

- **Author:** G3N-es.
- **Difficulty level:** Easy.

Qtractor doesn't have a Multi Track Ghost selection. In the MIDI editor, only one **"Ghost"** track can be selected for reference. However, we can achieve this multi-reference functionality by creating a specific track for "ghost" and saving it in our startup template.

The methodology is simple:

1. Copy the clips from the different tracks you want to track.
2. Paste them into your "Ghost" reference track.

When we copy and paste MIDI clips in Qtractor, they are linked (cloned, symbolic). This means that any changes you make to the original clip will automatically be reflected in the copies. If you edit the clips on the original tracks, the reference clip is updated. Qtractor also allows you to overlay clips on a single track.

This approach is fast and intuitive. It has advantages over selecting "Ghost" tracks on multiple tracks:

1. **Visually cleaner editing:** Show only the specific bars where reference is required, and not the entire track.
2. **More control:** You can create custom tracking clips, unlinking and editing them. So you're left with only the notes what do you want to follow.
3. **Multi-section references:** Imagine you want to use a reference to the **"Chorus"** for your **"Intro"** section. A multi-track ghost track doesn't allow you to do this. A dedicated **Ghost Track** does.

Conclusion

Qtractor doesn't have a Multi Track Ghost selection. Is it really necessary? You can achieve much more by creating a "Ghost Track" in your starter template.

How To - 17 Get Individual Drum Instruments with a MIDI Track

[How To - Contents](#)

- **Author:** G3N-es.
- **Difficulty level:** Intermediate.
- **Required Plugins :** [MIDI filters](#) ## Introduction Drums are multi-instrumental instruments (kick, snare, etc.). The most convenient way to sequence a MIDI drum kit is on a single track, as it allows access to all instruments from the MIDI editor.

This means that if you host a drum synth on that track, all instruments will share the same audio output. They cannot be mixed separately.

In many cases, this configuration will be sufficient. Since each instrument dominates at its specific frequencies, EQs focused on the spectrum of each instrument can give good results.

It's not all about frequencies. Sometimes we'll want to customize the envelope, color, reverb, and so on for a particular instrument. For example, shortening the release of an Open Hit Hat with a gate. Making a Snare bigger with reverb followed by a gate... In these cases, we'll need each instrument to have its own audio output.

There are multi-channel percussion plugins to tackle this task, but it's not the workflow we'll use here.

In this tutorial, we'll give you an alternative method. As with the multi-channel method, you can use GM or custom mappings.

We'll get the following advantages:

1. You don't need a multi-channel synth-plugin. You can use any synth-plugin (mono, stereo, multi-channel, etc.)
2. You can use different synths for each instrument.
3. You can use multiple synths to build a single instrument (see [How To - 15 Use Multiple SoftSynths on a Track](#)).
4. It's based on MIDI auxiliary sends, not audio ones. This simplifies routing.

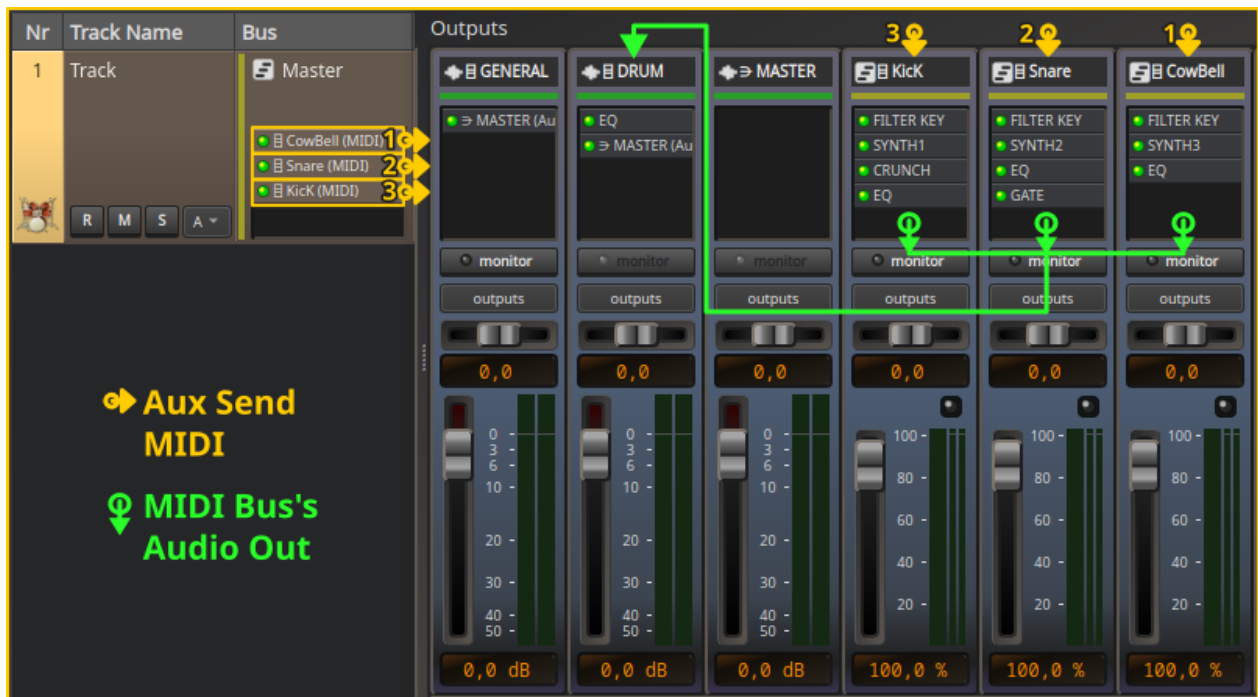
Workflow

We start with a MIDI drum track:

1. Create a MIDI bus for each instrument: Kick, Snare, Hit Hat.
2. Add a **MIDI Key-Range Filter** and a soft synth on each bus.
3. Edit each **MIDI Key-Range Filter** to assign the key (GM or custom) to each instrument.
Example: Kick, range 36 to 36.
4. On the drum track: Insert **Aux Sends (MIDI)** to the MIDI buses (Kick, Snare, Hit Hat).

You can now work with each instrument independently.

Example Schematic



In the schematic, we have also created a **Group Bus Audio** called **DRUM** to receive all the audio from the different drum instruments. This will be the audio output assigned to the MIDI instrument buses: Kick, Snare, Hit Hat.

This is just an example to show the mixing organization possibilities. It would have worked the same with the most basic configuration of a single **default MASTER** bus or with much more complex configurations.

How To - 18 Take Advantage of Qtractor's Hidden Tricks

[How To - Contents](#)

There are some small features in Qtractor that aren't well-known or documented. Let's discover them.

Define the Projects Directory

The Qtractor projects directory (the default location for projects) can be defined in templates.

Before saving the template, go to session properties: File > Properties.

In the "Directory" field, indicate the location you want to save your projects. From now on, every time you open Qtractor with that template, this will be the default directory for your projects.

Accessing Clips Below Other Clips

Qtractor allows a track to have overlapping clips.

In these cases, the clips above it prevent access to the background clips. To access them:

1. Set the "snap" to "beat."
2. Left-click and hold on the top clip.
3. Gently drag without moving the clip and release.

The top clip will move to the background, allowing you access. If there are multiple clips overlapping, repeat the process until you reach the one you want to edit.

Zoom Buttons

Want to zoom in as far as possible on a specific event or clip with a single click? Want to zoom out as far as possible for an overview with a single click?

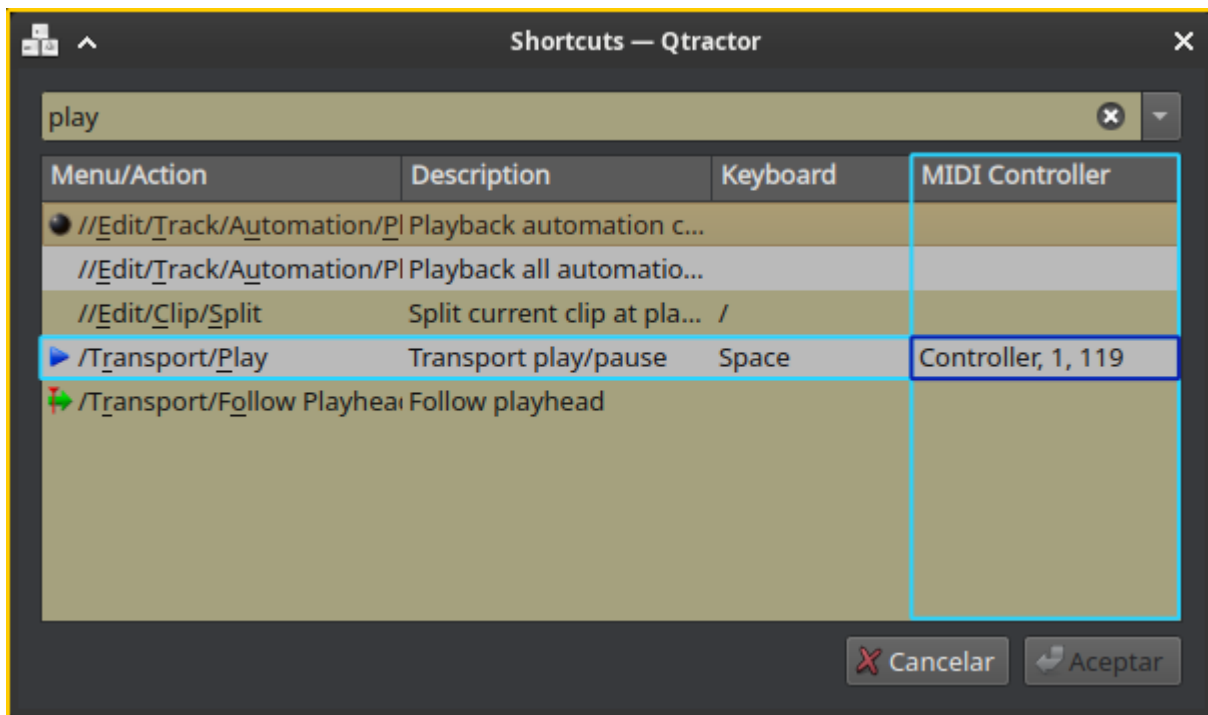
In both the sequencer and the MIDI editor:

- [**Ctrl**] + **Left Click** on $\ominus|\oplus$ buttons: \mp 100% Zoom
- [**Shift**] + **Left Click** on $\ominus|\oplus$ buttons: \mp 50% Zoom

MIDI Controller Shortcuts in the Sequencer

This allows you to trigger any Qtractor command from your MIDI controller. (Functionality not available in the MIDI editor.)

In the sequencer, go to: Help > Shortcuts.



Select an Entire Row of Events for a MIDI Note

If you click on a piano key and drag up or down, the entire row of events corresponding to the note is selected. If you continue dragging, all rows of the affected notes are selected. If you press the [Ctrl] key during the process, you deselect.

Quantize MIDI Recordings

Although it is not a recommended practice, quantizing MIDI recordings can be useful for certain tasks. The option is located in: View > Options > MIDI { Capture/Export: Quantize

Update the plugin list without restarting Qtractor

In the Plugins window:

- [Ctrl] + Left Click on the Rescan button

Precise Fast Forward and Rewind

In the “Time” toolbar, left-click on the seconds. Now scroll the mouse wheel. This also works with milliseconds, minutes, and hours.

Sliders (Faders on: tracks, buses, and plugin properties)

- Double-click or middle-click to restore the fader to its default position.
- Clicking and holding a specific point within the fader’s travel will move the fader uniformly to the click point.

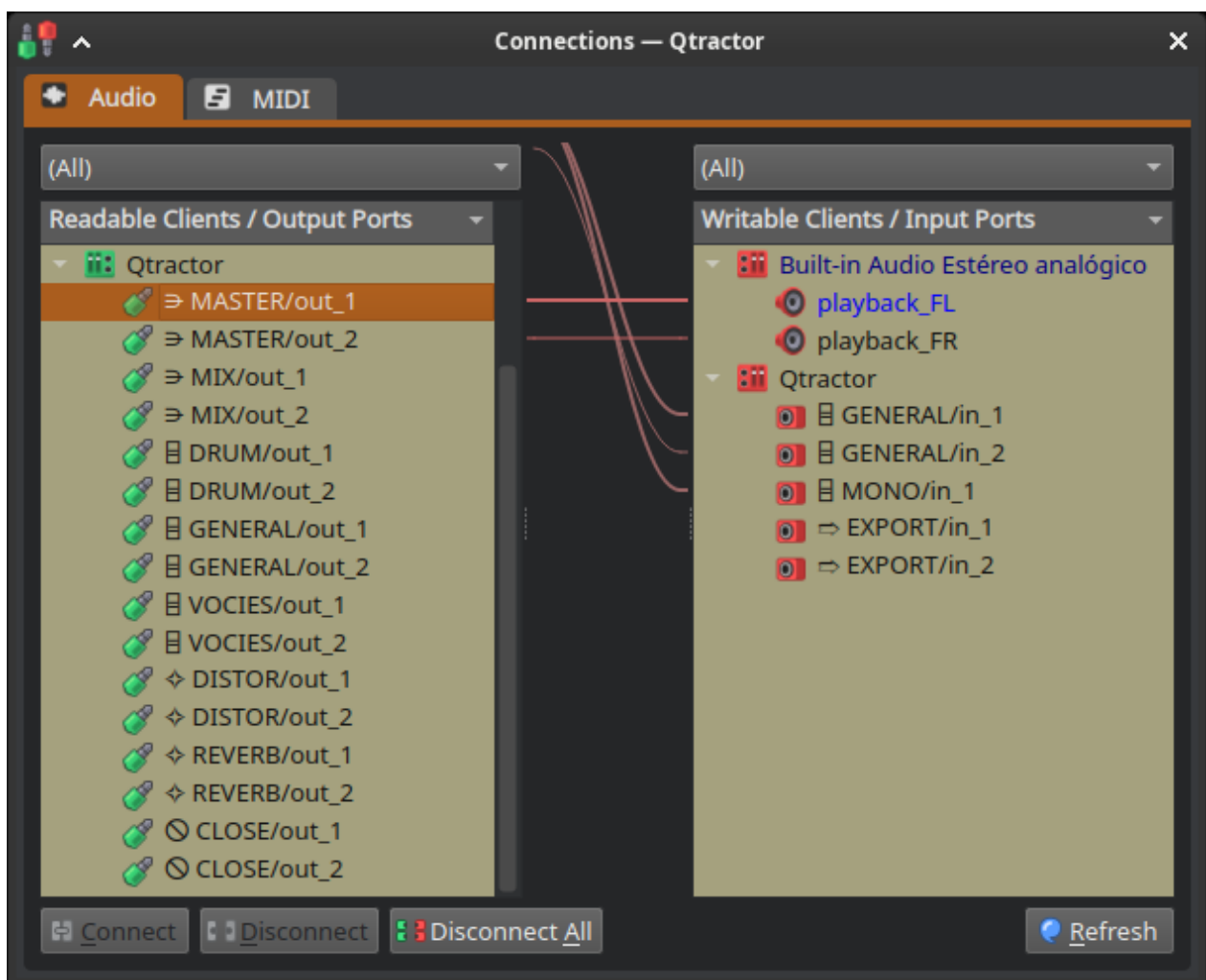
Emojis in Texts

Texts in Qtractor support unicode. This allows you to customize any name with icons: Tracks, Buses, Plugin Aliases, etc.

I find it useful for assigning icons to Bus functions so I can easily recognize them.

Examples of bus icons by function:

- 📁 Groups
- ⚡ Parallel Effects
- ➡ Mixer
- ➡ Bus intended for export
- Etc...

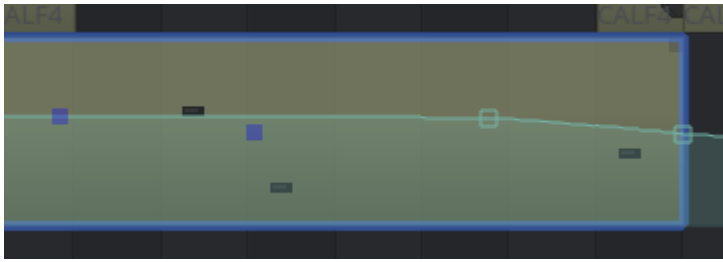


Copy/Paste Automation Points

On a selected track, with **Select Mode** set to **Automation**, use the mouse to lasso-select one or more automation points. Now issue a **Copy** and **Paste**. The newly displayed automation points can now be slid left or right (following whatever beat quantize value currently selected) and dropped into place with a left mouse click. Obviously, pressing **Esc** aborts the paste.

This approach is quite handy since many of the automation points we draw are actually the left-most side of a ramp or slope which matches the value of the preceding point. Of course, the value

of the approach scales when we want to duplicate complex patterns we've taken the time to design.



Use MIDI Tracks as Text Tracks

It's not a Qtractor feature, but it's a trick that can be useful. Add empty clips and edit their names with your annotations, sections, lyrics, etc.

Unofficial tutorials that are worthwhile

Tutorials

- [JACK Beginners Guide | by Conor Mc Cormack](#)
- [Qtractor QuickStart | by yPhil](#)
- [Sidechaining in Qtractor using Ducka | by suedwestlicht aka bluebell](#)
- [Qtractor Tutorial For Beginners, Step by Step Guide to Creating Your First Electronic Track | by rounakagag](#)

Video Tutorials

- [THE Qtractor Tutorial | by i-scores music](#)
- [Yassin Philip | Why, Phil? #LinuxAudio | by yPhil](#)
- [Sonoj 2019: A song in Qtractor from scratch | by Rui Nuno Capela \(rncbc\)](#)
- [Qtractor Tutorial and Review | by Demonic Sweaters](#)
- [Qtractor Tutorial Series | by rounakagag](#)

qtractor 0.3.0 manual pt1

Qtractor

Linux Audio & MIDI Multitrack Workstation

User Manual

Version 0.3.0

December, 2008

by

Rui Nuno Capela

James Laco Hines

Stephen Doonan

Table of Contents

1 Introduction

1.1 Abstract

1.2 Introduction

2 Installing and Configuring Qtractor

2.1 About Compiling Qtractor from its Source Code

2.2 Preparation—Required and Optional Prerequisites

2.2.1 Mandatory Software

2.2.2 Optional Support Libraries (at build time)

2.3 Downloading Qtractor

2.3.1 Qtractor for Everyone

2.3.2 Qtractor for the Experienced and Adventurous

2.4 Compiling and Installing Qtractor

2.4.1 Standard Compiling and Installation

2.4.2 Compiling for Native Linux VST Support (optional)

2.4.3 Compiling Qtractor with Debugging Code Included

2.5 Qtractor's Configuration Settings File

2.6 Audio and MIDI Input

3 Learning Qtractor—An Example Session

3.1 Preparation

3.2 Importing an Audio File

3.3 Connecting the MIDI data source to Qtractor

3.4 Creating a MIDI track

4 Qtractor—An Overview

4.1 Routing—Connections, Ports, Tracks and Buses

4.1.1 Routing—General Concepts and Information

4.1.2 Routing in Qtractor

4.1.3 Routing—Technical Notes

4.2 Qtractor's Main Window and Work Area

4.3 Understanding a Qtractor Session (recording or editing)

4.3.1 Session Audio Sample Rate

4.3.2 Session Properties including Time Signature and Tempo

4.3.3 Session Options

4.4 Files

4.5 Clips

4.5.1 Clip Summary

4.5.2 Audio Clip Properties

4.5.3 Clips and Tracks

4.6 Qtractor Main Workspace—Tracks Area

4.7 Mixer

4.8 Connections Window

4.9 Audio Effects Plug-ins

4.9.1 Summary

4.9.2 LADSPA

4.9.3 DSSI

4.9.4 VST (Linux Native)

4.10 MIDI Instruments

4.11 MIDI Editor

4.12 Audio / MIDI Export

4.13 Keyboard Shortcuts Editor

5 Qtractor Main Menu

5.1 File Menu

5.2 Edit Menu

5.3 Track Menu

5.4 View Menu

5.5 Transport Menu

5.6 Help Menu

6 Appendixes

6.1 References

6.2 Colophon

6.3 Contact Us

Index

1. Introduction

1.1. Abstract

Qtractor is a multi-track Audio and MIDI recorder and editor. The program is written in C++, and for the GUI (graphical user interface) elements, the Qt4 Toolkit and Qt Designer are used. Qtractor is free open-source software, licensed under the GPL, and the project welcomes all collaboration and review from the Linux audio developer and user community in particular, and the public in general.

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

The initial target OS platform is Linux, in which ALSA (Advanced Linux Sound Architecture [4]) and JACK (the Jack Audio Connection Kit [3]) form the supporting infrastructure for

recognizing sources of digital audio and MIDI (musical instrument digital interface) data, communicating with those sources and routing the data to and from various locations and programs (applications, including Qtractor) both inside and outside the computer and involving both software and hardware interfaces.

The goal is to develop Qtractor into a more and more full-featured and robust digital audio/MIDI workstation, especially appropriate for personal home recording studio use.

1.2. Introduction



Illustration 1.1: Main GUI window showing audio & MIDI tracks, Mixer & Connections windows

Although Qtractor will become more and more full-featured as it is developed, it can already be comfortably used by hobbyists as a personal home recording studio or “bedroom studio.” It can record, import, arrange and edit both digital audio and MIDI data. The functionality of Qtractor is contained within a graphical desktop environment that will be familiar to users of other popular multi-track recording/editing applications on any computer operating system, and follows the same design principles with many of the same or similar elements. In addition to recording digital audio and MIDI, Qtractor provides an environment for multi-track clip-oriented composing techniques common in modern music-making and aims to be intuitive and easy to use, yet powerful enough for the serious recording enthusiast.

Note: Qtractor is not what is known as a “tracker” type of audio/MIDI application, although it has the potential to function in that way if needed.

When used merely as an audio and/or MIDI recorder (a MIDI recorder was historically called a “sequencer”) or arranger, Qtractor is non-destructive, which means that the underlying files that contain the audio or MIDI data are not altered when those files are apparently cut into pieces, duplicated, pulled or pasted into a different order in time, or manipulated in any number of ways within the main Window (GUI interface) of Qtractor. However, when used as an audio or MIDI recorder, for example, or when editing previously recorded MIDI data in the dedicated MIDI editor, Qtractor’s actions can be destructive in the sense that newly recorded data (or altered MIDI data) replaces previously recorded data on the same track.

2. Installing and Configuring Qtractor

2.1. About Compiling Qtractor from its Source Code

If Qtractor is not available as a package for your particular type of Linux (a .deb package for Debian, Ubuntu or other Debian-based system or an .rpm package for Red Hat, Fedora, SUSE, etc.), then it must be compiled into an executable application from its source code (from the C++ programming code in which Qtractor is written) before it can be installed. In that case, one’s computer must have an appropriate compiler program installed (such as G++, the GNU C++ compiler) in order to compile the C++ source code of Qtractor.

For those who have experience compiling programs, the following preparation and instructions based on *autoconf* will be familiar. The process is fairly easy and straightforward, but both the process and the “build environment” (a collection of programs necessary for compiling) can be confusing for those who are not experienced, so those persons may wish to learn a little about the process first. There are many books and online resources such as Linux forums, email lists, wikis that can be consulted and used to learn how to compile and install programs’ source code.

2.2. Preparation–Required and Optional Prerequisites

In order to compile the source code of Qtractor to create an executable program, as well as to run Qtractor, some software must be installed (the mandatory software listed below) and some may be installed if the user wishes to enhance the abilities of Qtractor (the optional support libraries).

2.2.1. Mandatory Software

- **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/>

2.2.2. Optional Support Libraries (at build time)

If the functionality that these additional software libraries provides is desired for Qtractor, they must be installed before Qtractor itself is compiled from its source code.

- **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://breakfastquay.com/rubberband/>
- **librubberband** - Rubber Band Audio Time Stretcher, an audio time-stretching and pitch-shifting library <http://breakfastquay.com/rubberband/>
- **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/>

2.3. Downloading Qtractor

2.3.1. Qtractor for Everyone

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

<http://qtractor.sourceforge.net/>

2.3.2. Qtractor for the Experienced and Adventurous

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

At the command line (in a text terminal or terminal window) 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 (all in the same line):

```
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.

2.4. Compiling and Installing Qtractor

2.4.1. Standard Compiling and Installation

After downloading Qtractor and decompressing and extracting the archive if necessary (with the applications `gzip` and `tar`, for example), change directory in a command-line terminal to the resultant `qtractor` directory. Once inside the `qtractor` directory, type and enter the following command:

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

NOTE: To see all configuration options before entering the command sequence above, type

```
./configure -help
```

After typing the `configure` and `make` commands and waiting until the program has finished being compiled, become an administrator of your system (using either the `sudo` or `su` command to become, temporarily, the user “root”) and finish the installation by entering the following command:

```
make install
```

which will copy the `qtractor` binary executable (the Qtractor application or program) and associated desktop and icon files to common standard system locations.

2.4.2. Compiling for Native Linux VST Support (optional)

VST is a “Virtual Studio Technology” developed by Steinberg Media Technologies GmbH for their own proprietary audio and MIDI applications. VST support is not very easy to accomplish and is suggested only for experienced users. Because of licensing issues for this proprietary software, one must download the VST SDK (software developer’s kit) from the Steinberg Media Technologies GmbH website, specifically searching in the third-party developers section. It doesn’t matter whether you choose version 2.3 or 2.4 of VST, but choose one and only one. *Do not use VST 3.0. It will not work.*

In order to download the VST SDK zip-archive you will have to accept the license and supply some personal data, then download and unpack (de-compress and/or de-archive) the pertinent program header files, which are found in one of the directories listed below.

VST SDK 2.3:

Directory:

```
vstsdk2.3/source/common/
```

Files:

```
aeffectx.h
```

```
AEeffect.h
```

VST SDK 2.4:

Directory:

```
vstsdk2.4/plugininterfaces/vst2.x/
```

Files:

```
aeffectx.h
```

```
aeffect.h
```

Just copy the two files to somewhere else in your computer's directory structure. It is recommended that you copy those files into a standard "include" directory (eg. /usr/local/include or /usr/include), in which case all will be handled "automagically" by the ./configure build step. Otherwise you'll need to supply the path yourself, as in:

```
./configure --with-vst=/path/to/vstsdk2.x/include
```

Once Qtractor is properly compiled, you will probably want to download some native VST plugins. But Qtractor must be told where it can find these VST plugins. To accomplish this, currently you'll need to set (create and assign a value to) a variable known as an "environment variable." The variable must be named VST_PATH and the value assigned to this environment variable must be the path (location on the computer) where the VST plugins have been placed. You can do this by entering the command (in a command-line terminal or terminal window):

```
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>

2.4.3. Compiling Qtractor with Debugging Code Included

Although Qtractor is a mostly stable program, there could be problems that eventually show up. This short guide will explain to you how to build Qtractor with debugging code built in, making it easier to locate where the code exhibits problems.

Rebuild it all from scratch, with:

```
./configure --enable-debug && make
```

Enable core dumps in a shell session:

```
ulimit -c unlimited
```

From the same shell command line, run the program until it crashes. You'll see something like this in the output when it happens:

```
Segmentation fault (core dumped)
```

Locate the dumped core file. Depending on your environmental settings it might be just named core or something like core.1234 (1234 is the process-id number of the crashing program) located on the last directory the program was current.

Load the core dump file into gdb:


```
gdb ./qtractor /path/to/core
```

At the gdb prompt just enter:

```
gdb> bt
```

or:

```
gdb> thread apply all bt
```

2.5. Qtractor's Configuration Settings File

Qtractor keeps a separate set of its run-time settings and configuration state (a “memory” of certain settings) for each person who uses Qtractor, in a file located in the user's home directory as in the following example (on your computer the word “user” in the example would be replaced with your own user name):

```
/home/user/.config/rncbc.org/Qtractor.conf
```

Normally, there is no need to edit this file because it is recreated and rewritten every time Qtractor is run.

2.6. Audio and MIDI Input

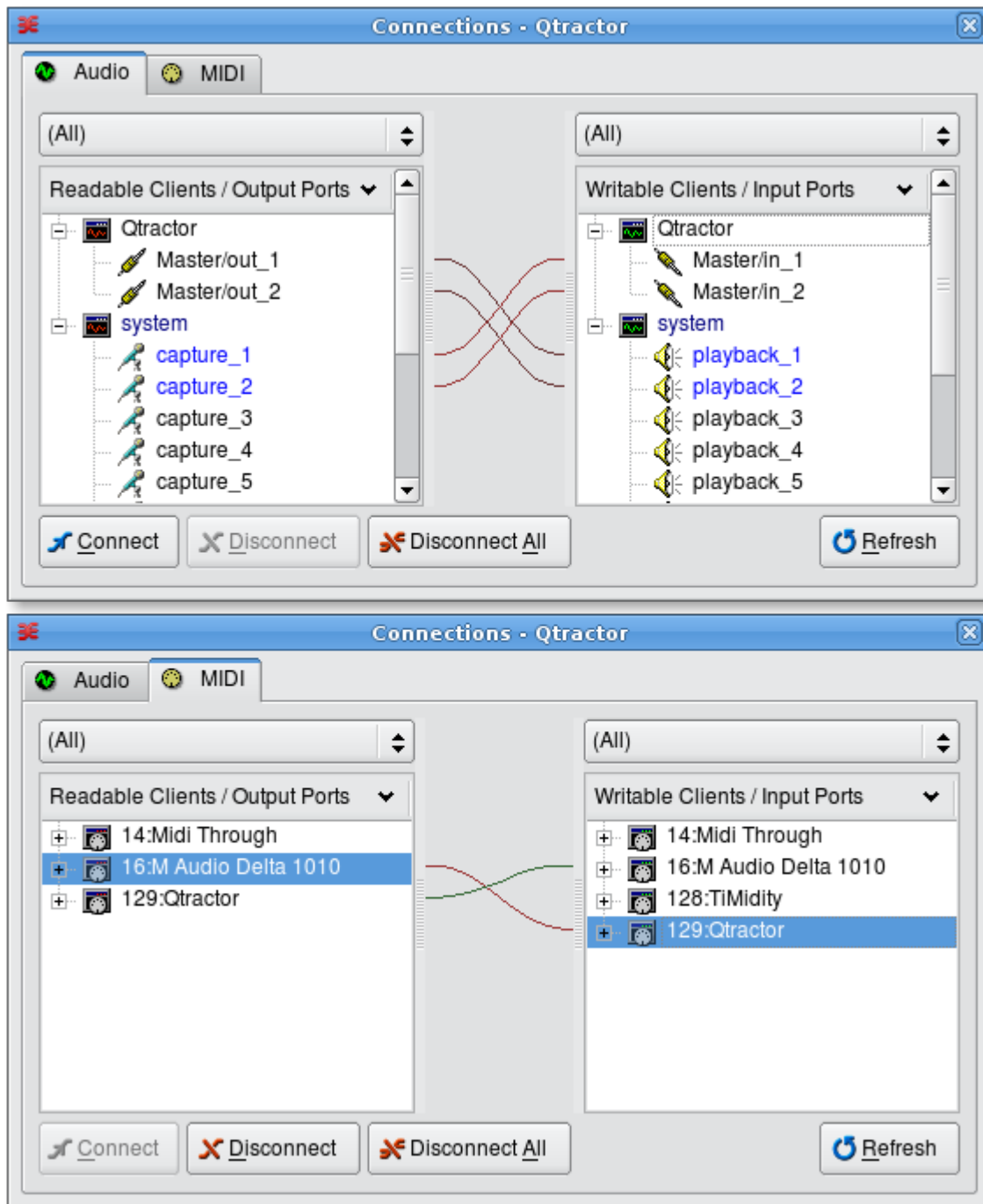


Illustration 2.1: Qtractor’s Connections window, showing both the Audio and MIDI tabs; “readable” ports are sources of data (where audio or MIDI data can come from) while “writable” ports are places that data can be routed to (sent to).

Qtractor can record both digital audio and MIDI data, but it does not know what audio or MIDI data you wish to record nor does Qtractor automatically make any connections to sources for that data. Instead, you must route audio and MIDI data to Qtractor manually, like building a pipeline from the source of the audio or MIDI to Qtractor. Qtractor includes a utility for doing this: the Connections window, pictured below. But Qtractor and its Connections utility depend upon the supporting software infrastructure mentioned previously, composed of ALSA and Jack, in order to both recognize sources of data and to receive data from those sources. ALSA is responsible for

knowing about and communicating with audio and MIDI hardware and some software, and Jack is used for routing audio and MIDI data to and from various hardware and software “ports” within the computer or attached to it. Both ALSA and Jack, working together, make it possible to route audio and MIDI data to Qtractor. You simply must remember to connect at least one source of that data to Qtractor first, before you can record some of that data in Qtractor.

3. Learning Qtractor—An Example Session

This chapter is written for those who may not be very familiar with digital audio recording or MIDI “sequencing” applications, or who wish to gain a quick overview of how Qtractor works and can be used before exploring the program in greater depth and learning its features in detail. It describes an example Qtractor session and serves as a walk-through of the program. The reader can follow the writer as he creates a Qtractor session and uses Qtractor to record, import and edit MIDI and audio data.

3.1. Preparation

You’ll use Qtractor to record or import several tracks of MIDI and audio data. A MIDI-triggered tone generator (in this case, a rack-mounted tone generator outside the computer, although it could just as easily be a “soft synth” inside the computer) will produce the sound for the MIDI parts as Qtractor plays them back. The audio tracks will be either recorded from some external source, or pre-recorded audio files will be imported into, played back and edited in Qtractor. A final “mix down” audio track in Qtractor will contain the combined audio result of playing back and simultaneously recording the other audio and MIDI tracks.

3.2. Importing an Audio File

The first thing you would like to do is to import a pre-recorded audio file of drums and other percussion, to form the basis of the rest of the recording session.

3.3. Connecting the MIDI data source to Qtractor



Your MIDI piano-like keyboard normally routes its MIDI data (created when you strike the keys, for example) to its own internal tone generator, which then sounds like you’re playing a real piano, or electric piano, harpsichord, bass guitar, etc. However, for this project in Qtractor you want to route the external keyboard’s MIDI data to Qtractor. So, using a standard MIDI cable you connect the keyboard’s MIDI output to the MIDI input of your sound card, and inside the computer you will route the MIDI data from the sound card to a Qtractor MIDI input bus.

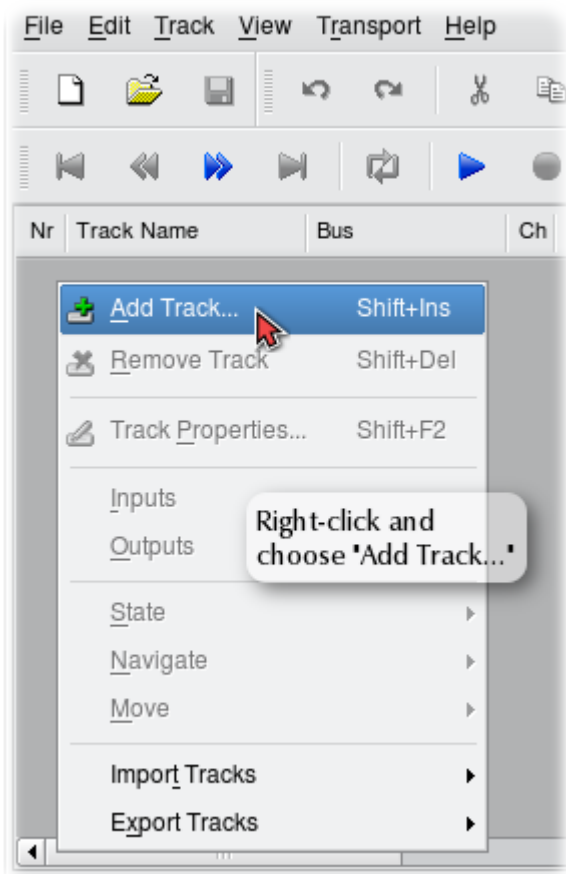
After connecting the MIDI cables, launch Qtractor, which also launches the JACK daemon (jackd, part of the Jack Audio Connection Kit mentioned previously) if jackd was not already running. Then open Qtractor’s Connections window, pictured in Illustration 2.1, page 9. In the Connections window MIDI tab, connect the MIDI output of the sound card (in the left pane, marked “Readable Clients / Output Ports”) to Qtractor’s listing in the right pane of the window (marked “Writable Clients / Input Ports”). One can connect an output port (the source of the data, MIDI or audio) to an input port (the port that will receive the data), in several ways: one is to highlight a port in the list in the left pane, highlight a port in the right pane, then right-click (third-button-click if a person is left handed and uses a mouse in a reverse configuration) and from the pop-up menu select “Connect;” another way is to click a port on the left side and with

the mouse button held down, “drag” the mouse to a port in the list at the right until the port is highlighted, then release the mouse button. Using either method, a line representing a “virtual cable” will appear between the two ports in the middle section of the window.

Then close Qtractor’s Connections window by clicking its button near the top of Qtractor’s main window and workspace.

3.4. Creating a MIDI track

Now that Qtractor can receive MIDI data, it’s time to create the first track in order to record that data. Right-click in the blank pane at the left in Qtractor’s main window and from the popup menu choose “Add track...,” (or you could choose the menu item **Track -> Add Track...**).



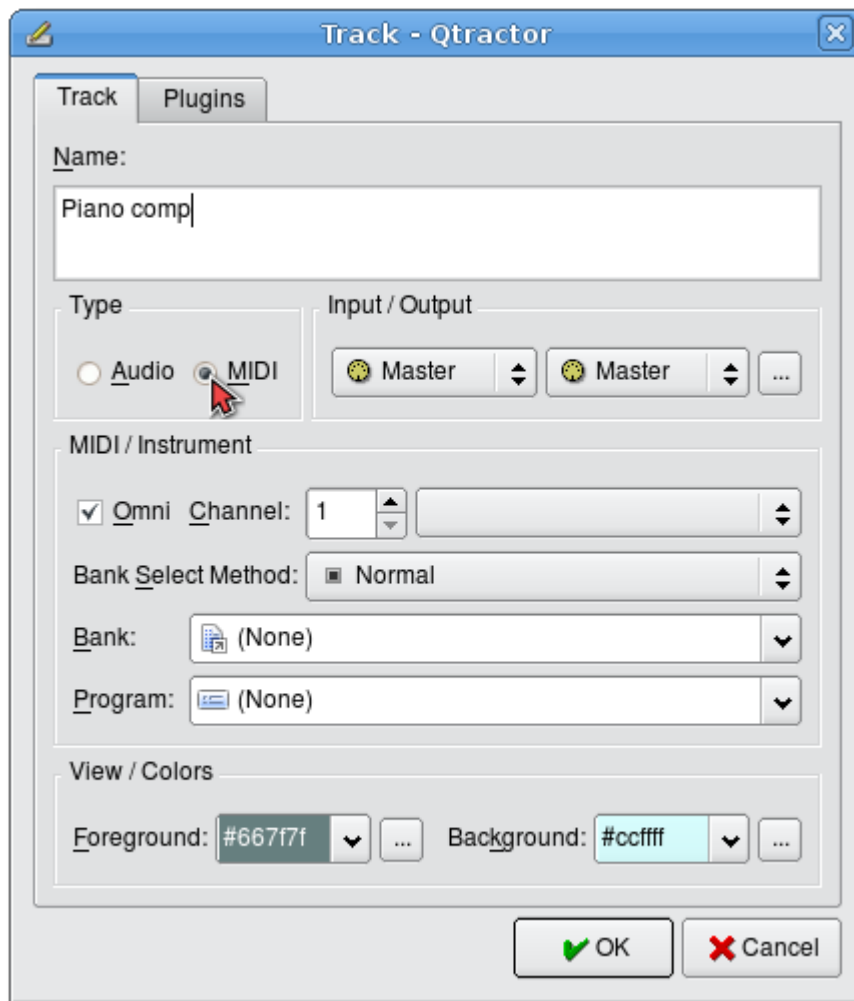


Illustration 3.1: Track Properties window

Qtractor’s Track Properties window opens.

In the Track Properties window, click the MIDI radio button: you want this track to be used for MIDI rather than audio data. As soon as you click the MIDI button, Qtractor automatically connects the track to Qtractor’s master MIDI input and output busses. These can be changed by using the drop-down lists and the ellipsis button in the Input/Output area of the Track Properties window, but there is no reason right now to do so. You can replace the default name “Track 1” with, for instance, “Piano comp.”

Leave the MIDI Channel at the default “1,” although this really makes no difference at the moment because this channel designation is mainly for output. That is to say, when the track is played back and its MIDI data is sent to somewhere outside Qtractor (like to a tone generator or synthesizer) that data will be sent on the MIDI channel specified here (there are 16 possible channels). But you’re going to record now, so the eventual output MIDI channel doesn’t matter and can easily be changed later, after you have recorded all the MIDI tracks you wish to record, and before you play them back.

What *does* matter however is the **Omni** checkbox. This setting determines whether Qtractor will record MIDI information received on *any* MIDI channel (omni means all), or whether (if *_un_checked*) will record only the MIDI data it receives on the same MIDI channel number as the output channel number displayed just to the right of the Omni checkbox. Your MIDI keyboard might be set to transmit MIDI data on only one channel (e.g., MIDI channel 1), so you have to make sure that each of the MIDI tracks you plan to record will receive data from any

MIDI channel (which will of course include the MIDI channel your MIDI keyboard is transmitting on) by selecting the Omni checkbox, despite the fact that all MIDI tracks will later be set to *send* their MIDI data each on a different channel (channels 1, 2, 3, etc.). Click the “OK” button and a new empty track appears in Qtractor’s main window.

Now is a good time to save this new session, so choose the menu item **File -> Save....** This opens the Session Properties window, Illustration 4.4, page 17. In the Session tab of the Session Properties window, give the project (session) a name (Qtractor will append the filename extension .qtr (or .qts) to this name to indicate a Qtractor session file), and you may choose to create a directory for Qtractor to store the MIDI and audio files that will become a part of this project or session. Give the directory you create a name similar to the session name.

Now that the session has been saved, open the Session Properties window again using the menu item **File -> Properties**. This time click the Properties tab of the window and set both the time signature (roughly, how many beats per measure) and the tempo so that you can use the metronome to help you keep the MIDI parts synchronized as you record them.

You need to arm or set the track ready for recording, by clicking on the “record” button for that track. That button is the “R” button that is present on every track strip, besides the other “M” and “S”, respectively for “Mute” and “Solo” track state settings. You can also set the record ready state for the current selected track by checking the menu item **Track -> State -> Record**. Either way the “R” button will turn red and the track is then set armed and ready for recording.

Now it’s time to turn on session recording mode. This is accomplished by clicking on the Record button in the main transport tool-bar (big red circle). You can do the same action through the menu item **Transport -> Record**.

Now you’re ready to record your first MIDI track. Take a little breath and... press “Play”, the space-bar or **Transport -> Play** menu item: you’re rolling. Hit the MIDI keyboard and see a brand new MIDI clip taking shape while you’re performing.

When done, press “Play” again to stop. Save your session and enjoy.

(continues on [qtractor 0.3.0 manual pt2](#))

qtractor 0.3.0 manual pt2

(continued from [qtractor 0.3.0 manual pt1](#))

4. Qtractor—An Overview

4.1. Routing—Connections, Ports, Tracks and Buses

4.1.1. Routing—General Concepts and Information

Qtractor can record and play digital data, specifically digital audio and MIDI data. To record, Qtractor must *get* the audio or MIDI data from somewhere, and to play that audio or MIDI data Qtractor must *send* it to somewhere that is capable of understanding an audio or MIDI data stream and producing (for example) some sound. The process of directing such digital data to and from— and through—certain software and hardware is called routing, which is simply choosing the route that the data will take. Not much routing is done automatically; most routing is left to the user, because it is the user’s preferences, desires and goals in any specific project that in large part determine how the audio and MIDI data should be routed.

In order to route data, one must have a knowledge and understanding of the various connections and pathways that are available and through which data can flow or be deposited. A *bus* is like a pipeline through which data can travel. Often, more than one stream of data can flow through a single bus, side by side so to speak. A *port* is like a valve at one or both ends of a bus. The valve is normally closed, not allowing any data to flow through unless a connection, such as another bus, is made to the other side of the valve. In that case the valve (port) opens to allow data to pass through it from one bus or pipeline into another. However, simply because a connection is made and the port is open doesn’t mean that data is flowing through the available port and buses. The data flow in either direction is initiated by the software or hardware at one or both of the ends of the buses. Often, the data in buses can travel in only one direction. Sometimes it can travel in both directions (duplex mode).

Qtractor has input buses (both audio and MIDI) to which “data streams” from other software or hardware can be connected, and output buses which can be connected to the ports and buses of other software or hardware both inside and outside the computer, using routing procedures explained in this manual.

A track can be thought of as a place where digital audio or MIDI data is deposited rather than a pathway through which such data moves. The data is placed into a track by a recording or paste or import procedure, for example. Despite this, tracks are a part of the signal or data flow, the pathway that audio or MIDI data can take through Qtractor and out of Qtractor, via a bus, to other software or to a hardware port on the sound card installed into one’s computer, for example. The reason that a track is considered part of the signal or data flow is that a track can be examined (by Qtractor) in a sequential fashion to “pluck” a *copy* of the data from the track and send that data to one or more buses and through those buses to other destinations. During recording, a track becomes the *destination* for audio or MIDI data coming into Qtractor’s input bus(es). During playback, a track becomes the *source* of audio or MIDI data that is then sent through a Qtractor output bus to a destination such as to the sound card and its attached speakers.

Routing is very flexible and therefore signal/data flow can become very complex. Data streams can merge and flow through the same bus for a while, or split and go through separate buses to

different destinations, or even through different buses to the same destination, meanwhile passing through separate and different intermediate software or hardware environments in which the data is possibly manipulated or altered in some way. Because of this complexity it is beneficial to develop a solid understanding of the issues related to routing and the many possible routes or pathways that data can take, and the many possible connections that can be made, all at the user's discretion.

Below is a representation of the connections (ports) buses and routes available in Qtractor through which audio or MIDI data can flow.

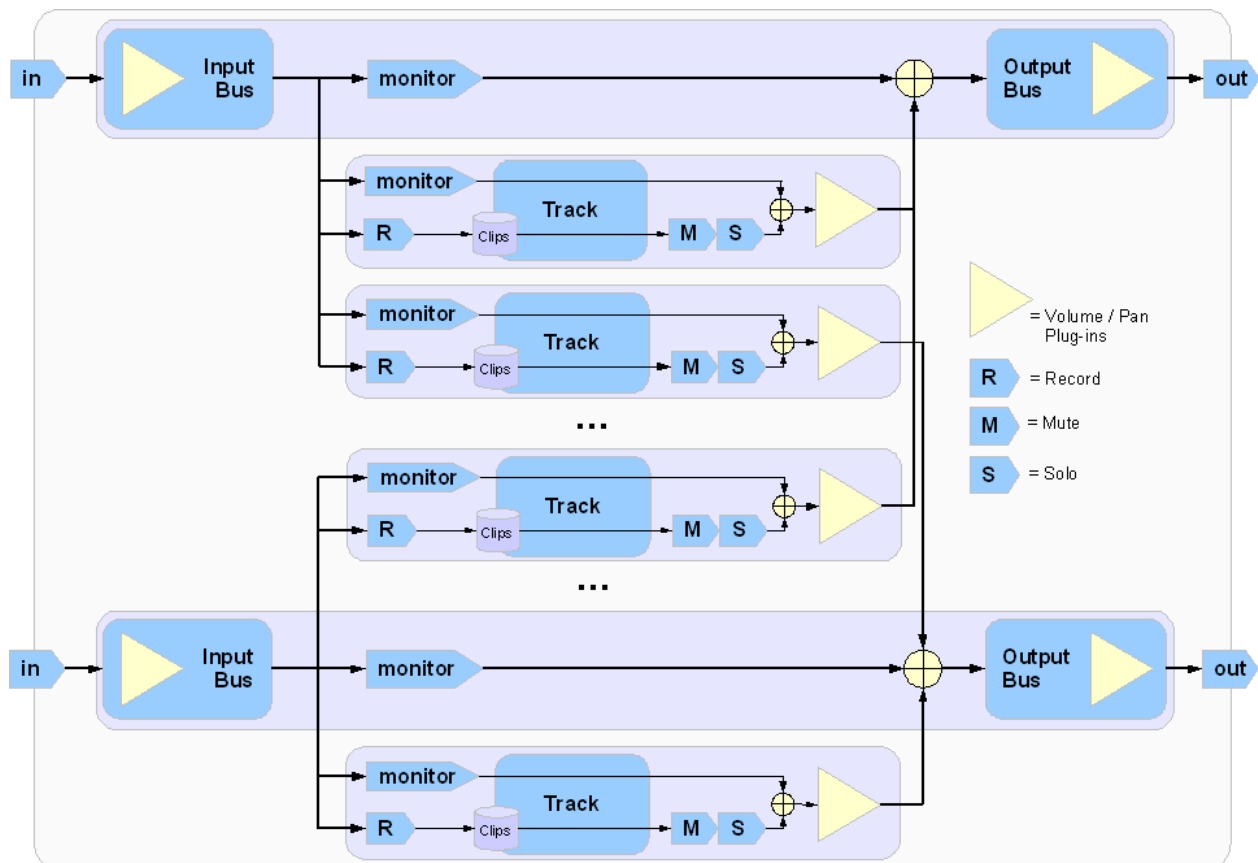


Illustration 4.1: Audio & MIDI data flow, in, through and out of Qtractor

4.1.2. Routing in Qtractor

Routing in Qtractor is accomplished in several ways including:

- using the Connections window (Illustration 2.1) to connect Qtractor's input and output buses to outside *sources* (for “reading” data in order to record it, for example) and *destinations* (for “writing” data to an outside destination in order to play back and listen to it, for example);
- in the Track Properties window (reference to illustration goes here...) where input and output buses are assigned to each track;
- changing the “state” of each track (there are four states, explained in detail later: record, mute, solo and monitor, some of which are mutually exclusive and others which are not);
- using the buttons “R” (record), “M” (mute) and “S” (solo) which are visible within each track's entry in Qtractor's main window;

- adding plugins to tracks or buses in the Mixer window or **View -> Buses** window.

4.1.3. Routing–Technical Notes

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. Illustration 4.1, page 14 shows one typical signal flow block diagram.

4.2. Qtractor's Main Window and Work Area

Qtractor's graphical user interface follows a standard design of most modern digital audio and MIDI workstations. The interface is easy and intuitive enough to easily interact with in order to discover the potential of the underlying inner core of the application where its functionality is implemented.

The following illustration shows an overall view of the GUI with an example session loaded into the workspace.

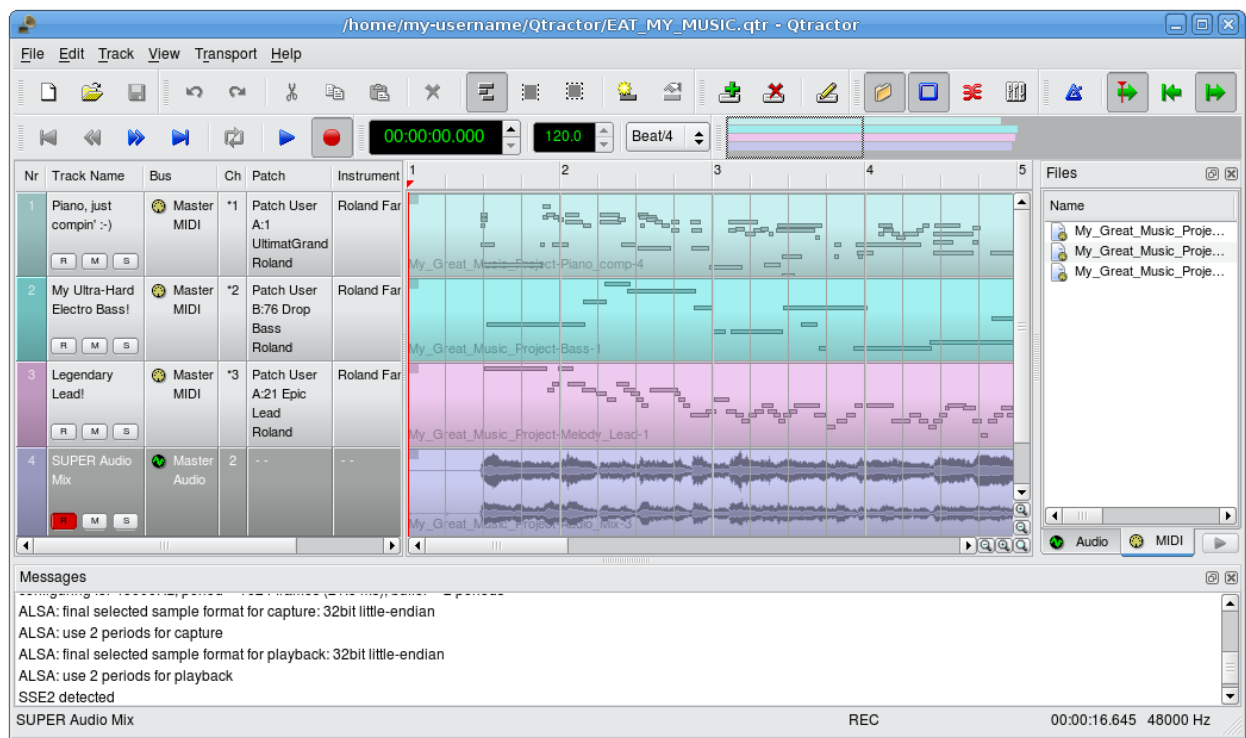


Illustration 4.2: Qtractor’s main window and work area

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

- Menu and Tool-bars at the top
- Track list and information at the left
- Track (audio and MIDI) data view representation and file list on the right, with time scale above.

The track data view section in the middle is where most of the action takes place. It contains visual displays and representations of audio waveforms or MIDI data. This section is used for editing Clip objects (portions or all of any particular audio or MIDI file, recorded or imported) and for navigation within the project or “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 (the patch-bay), 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*, available from the View menu in the main menu bar.

4.3. Understanding a Qtractor Session (recording or editing)

A Qtractor session project contains all the information about all your Clip Objects, placement of Clip Objects, Mixer setup, plug-ins, tempo, time signature and Connections patch-bay. When creating or saving a project, all this, and any related settings are saved on your hard disk within this session project file.

4.3.1. Session Audio Sample Rate

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. Real-time sample rate conversion is also going to use quite a bit more valuable CPU resources.

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 plug-ins, set the proper sample rate, etc. Until this feature is available, please follow the recommendations listed above.

4.3.2. Session Properties including Time Signature and Tempo

To access the session properties, choose the **File / Properties** menu item. Here, you can name your Qtractor session, set the tempo, time signature (how many beats per bar) and decide how many “ticks” (the smallest time unit in a session) will be within the time span of each beat.

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.

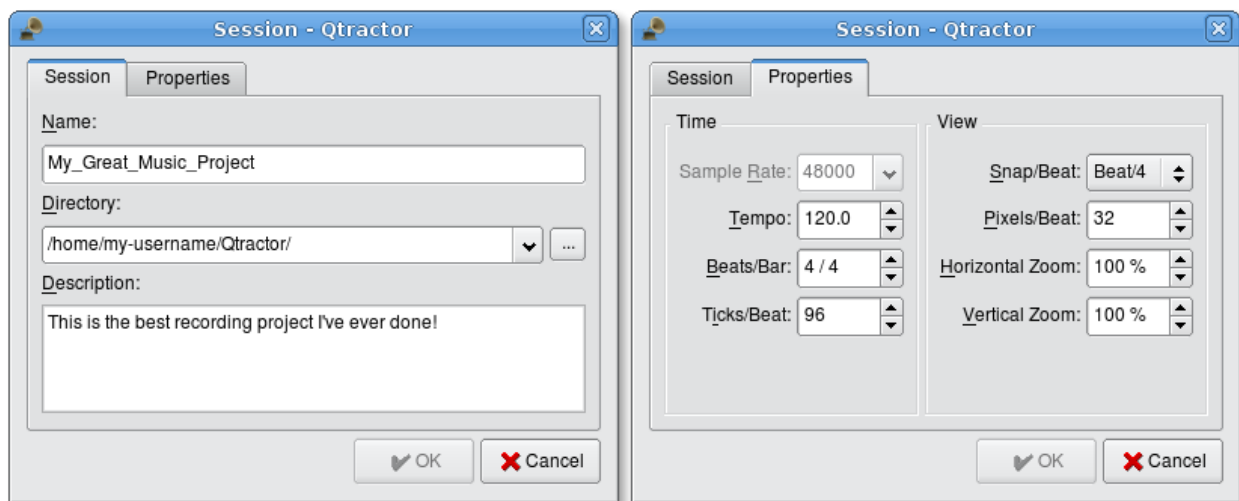


Illustration 4.3: Session properties window showing both Session and Properties tabs

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.

4.3.3. Session Options

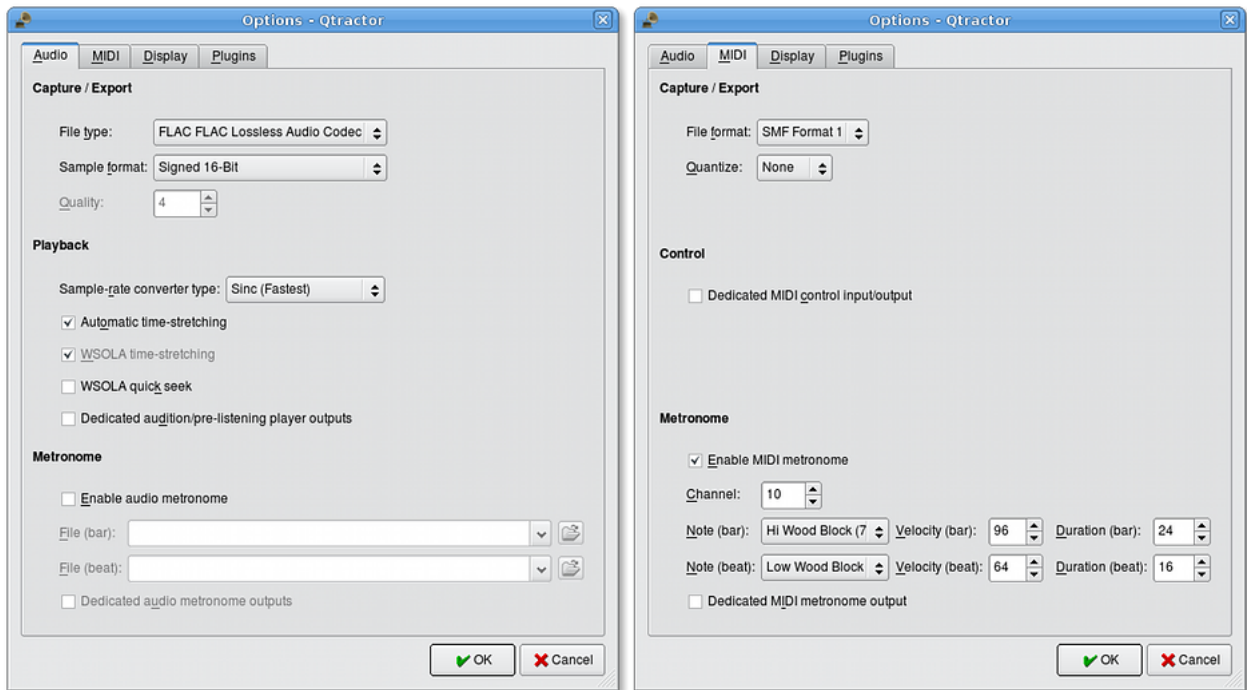


Illustration 4.4: Session Options window, showing both Audio and MIDI tabs (sections)

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 are the default settings which are not saved within your session file.

4.4. Files

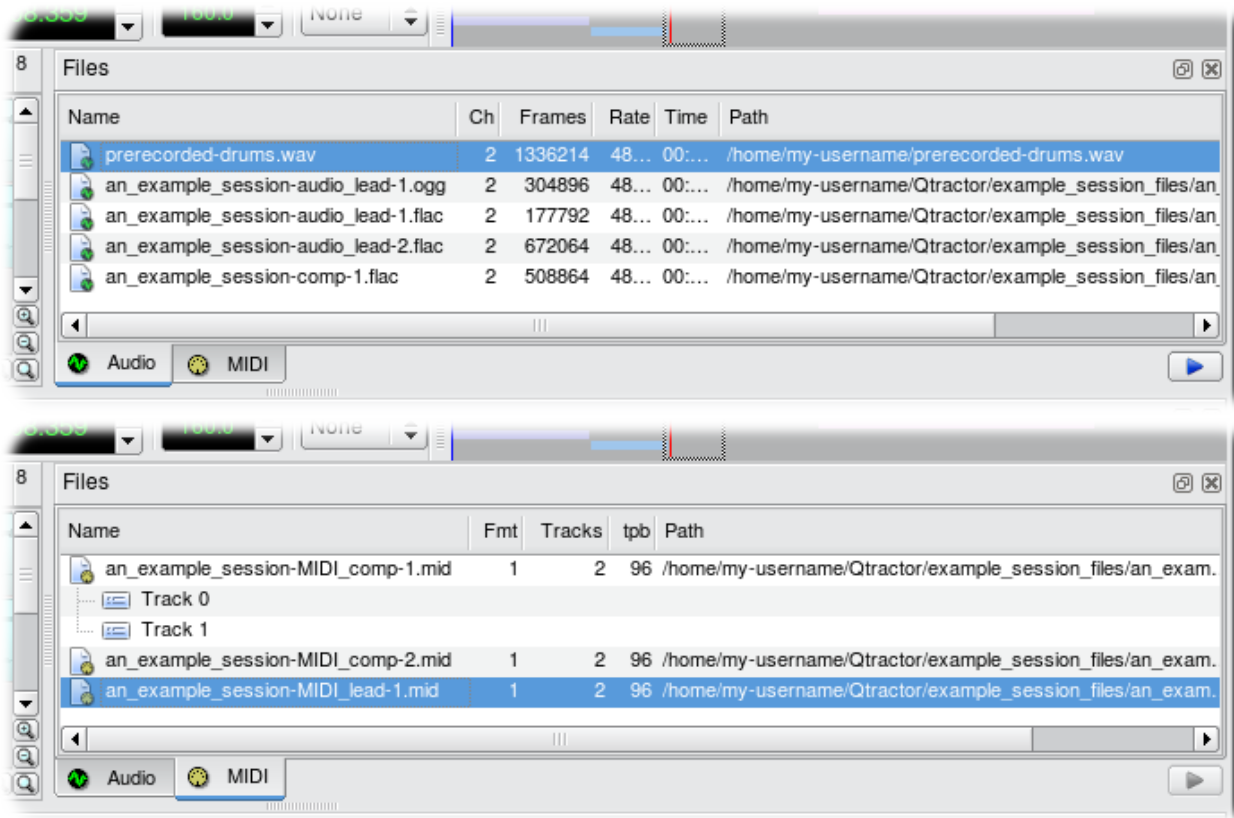


Illustration 4.5: The Files pane of Qtractor’s main workspace; both Audio and MIDI tabs are shown

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 imported using the menu command **Track / Import Tracks / Audio...** or **MIDI...**, or by using the context menu (“right-click” menu) in either the Audio or MIDI tabs of the Files pane and choosing **Add Files...**. 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, homebrew implementation.

4.5. Clips

4.5.1. Clip Summary

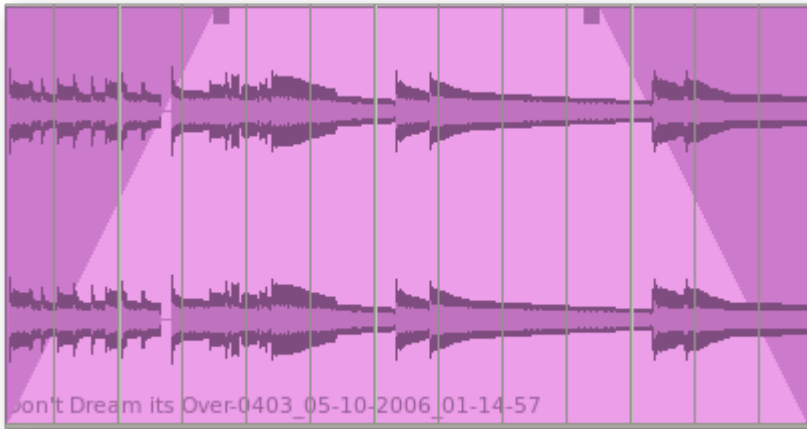


Illustration 4.6: An audio clip, with left and right upper corner handles drawn inward to create a fade-in and fade-out

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.

4.5.2. Audio Clip Properties

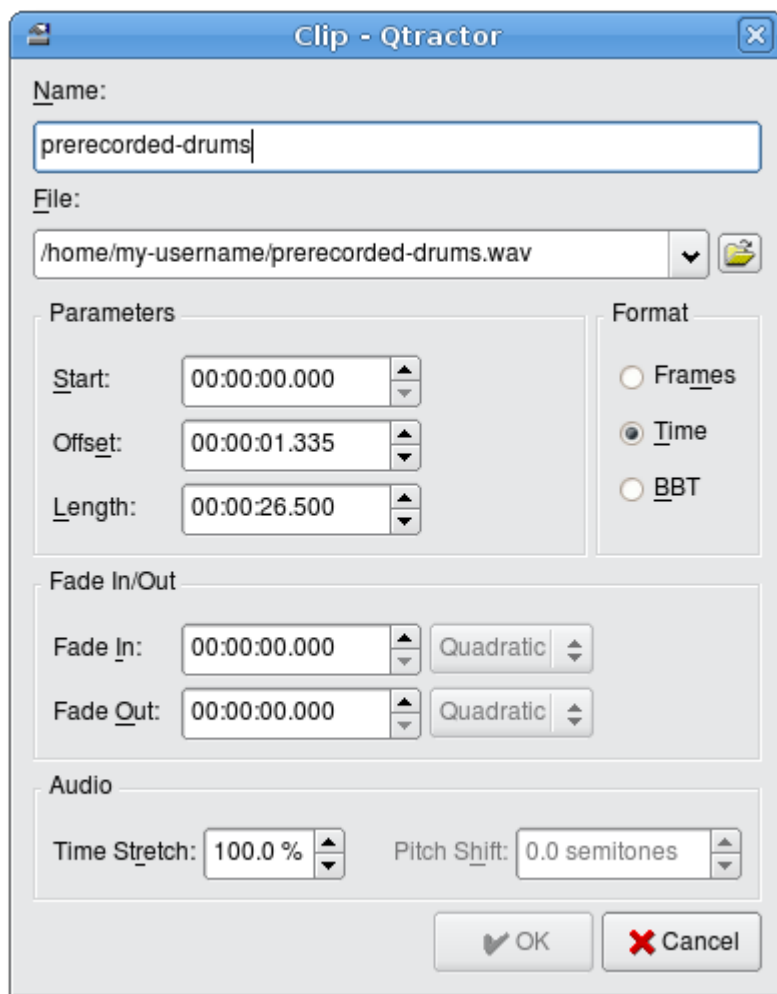


Illustration 4.7: Audio clip properties window, which appears with a double- click or right-click -> Clip -> Edit... on an audio clip's visual representation in Qtractor's main workspace

Clip properties include its label (Name), File path, start time (location), offset and length (in frames), fade-in and fade-out length (in frames) and time 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.

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. Empty clips may also be created by right clicking on a track, and choosing **Edit / Clip / New....** 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. A Clip may be split by positioning the playback head where you want to split it, right click the Clip, and choose **Edit / Clip / Split**. If you use this often, it is far more convenient to assign a keyboard shortcut to accomplish this task.

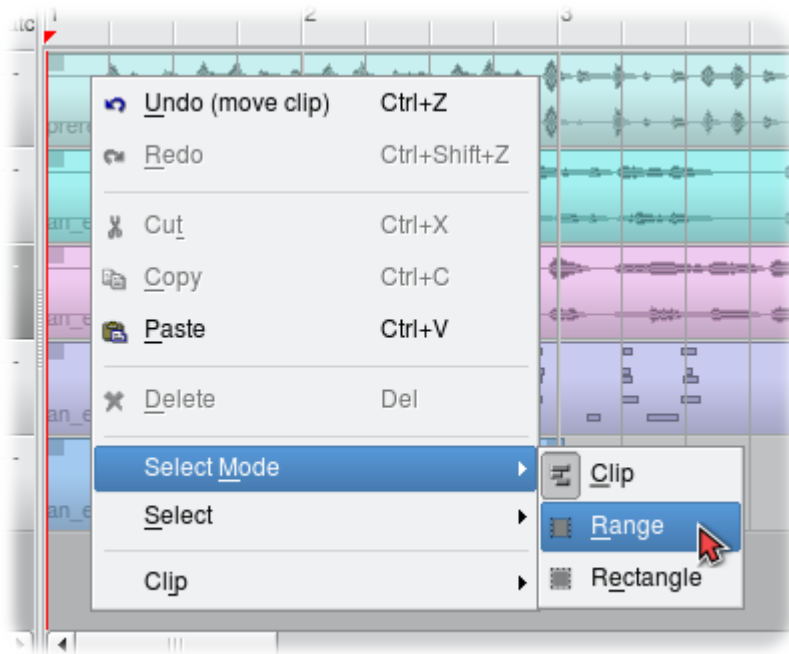


Illustration 4.8: Select Mode context menu (right-click menu)

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 (**Edit / Select Mode**).

- 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

4.5.3. Clips and Tracks



Illustration 4.9: Mute, Solo & Record buttons for individual 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)

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.

4.6. Qtractor Main Workspace–Tracks Area

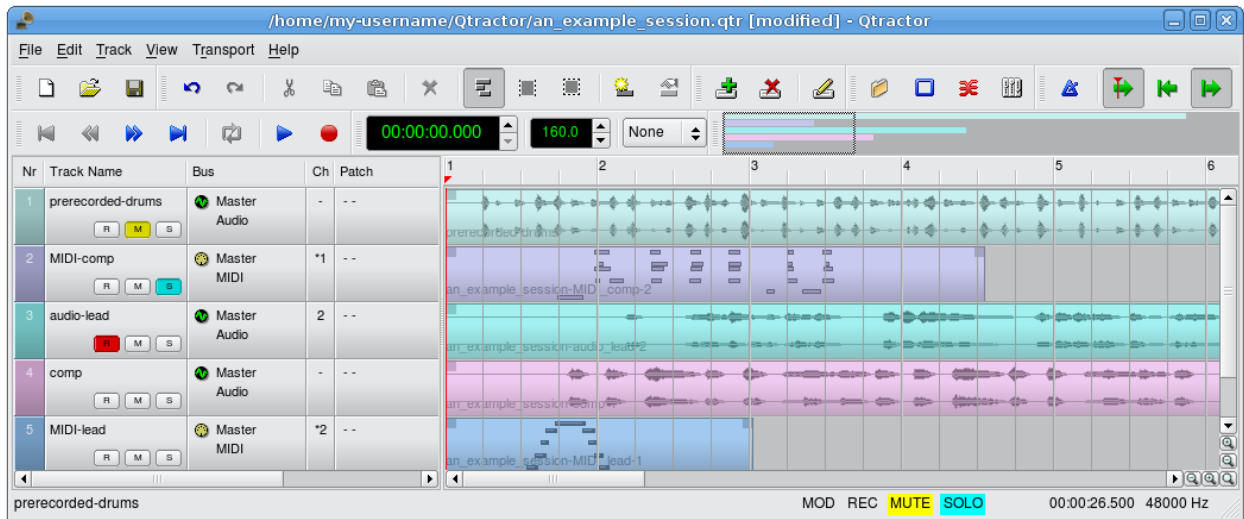


Illustration 4.10: Audio and MIDI tracks in the main workspace of Qtractor

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 tracks area of the main workspace window.

The tracks area has two panes, the left one displays the list of tracks with their respective properties and the center-right pane is the main tracks view canvas window where 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.

4.7. Mixer

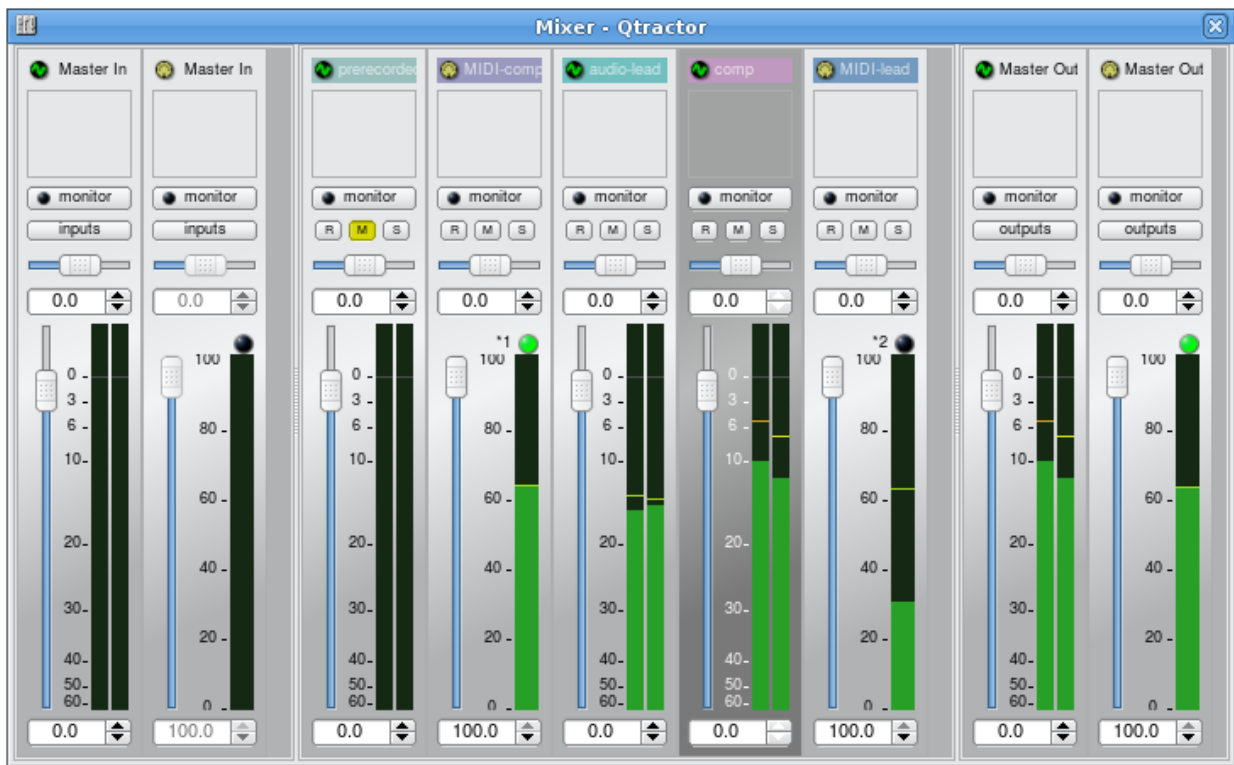


Illustration 4.11: Mixer window

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.

4.8. Connections Window

The Connections window (Illustration 2.1, page 9) 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.

4.9. Audio Effects Plug-ins

4.9.1. Summary

There are three types of plug-ins supported within Qtractor. LADSPA, DSSI and VST. Plug-in support is available for all audio input and output buses and for all audio tracks. All plug-in 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.

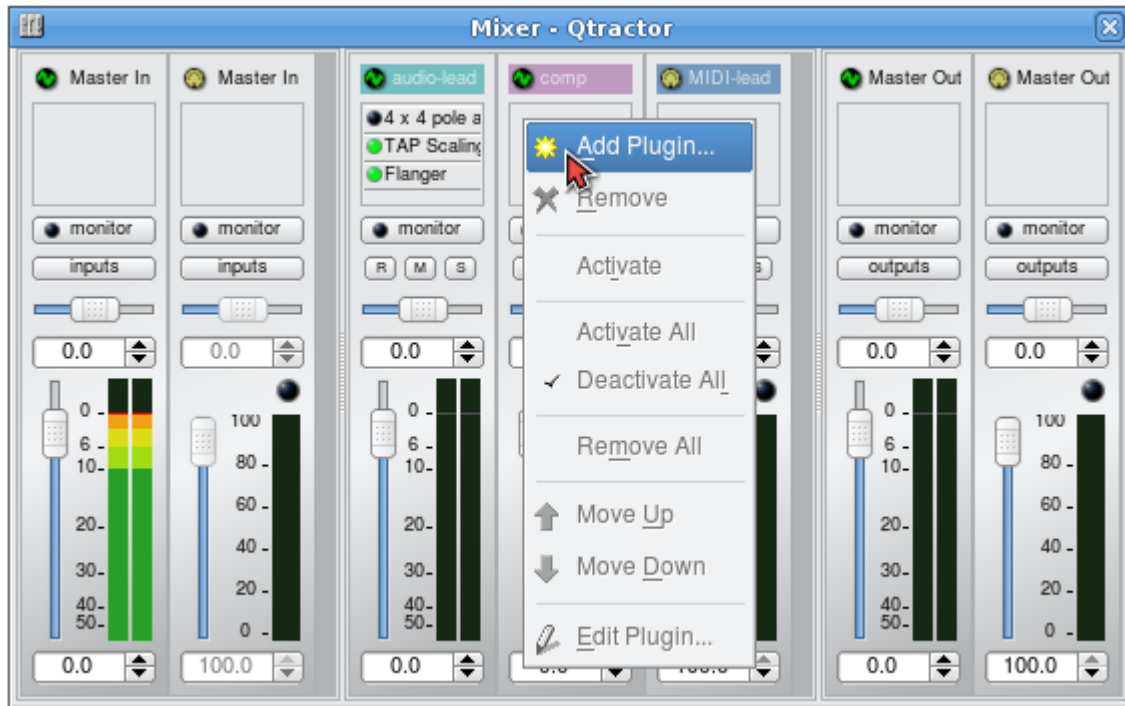


Illustration 4.12: Mixer window showing audio plug-in area at top; plug-ins are added, removed, activated/deactivated, moved in the processing chain and their settings edited using a right-click context menu.

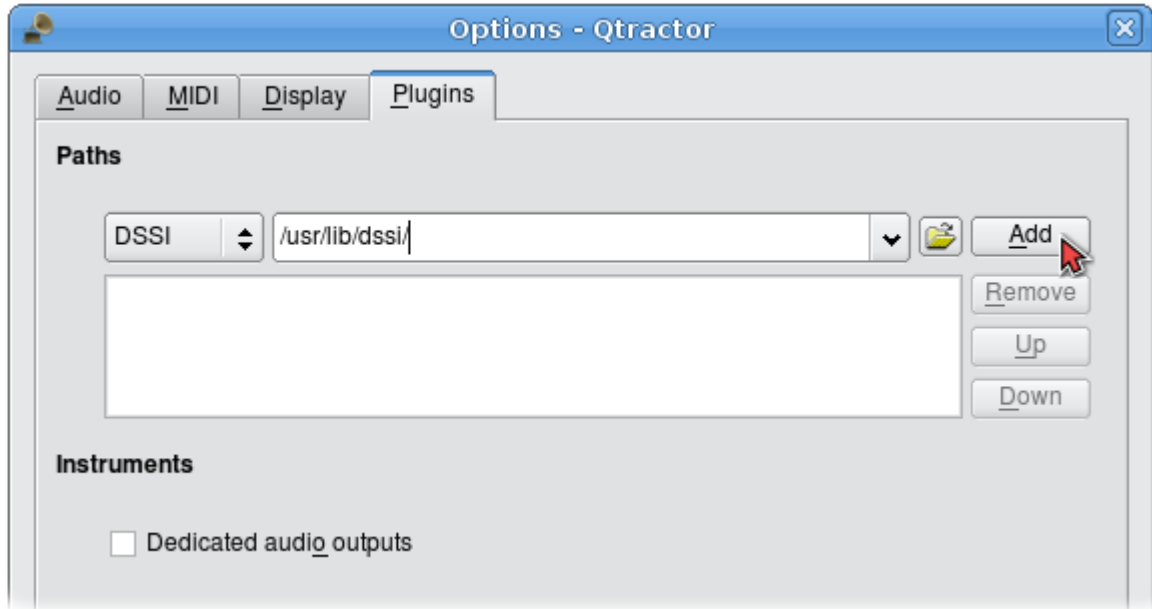
Also, you may drag-and-drop all plug-in instances over the mixer strip channels. You can move, drag and drop inside the same strip or over to, and from any other. You may also copy plug-ins from one channel strip to another as well. A mini menu will ask if you want to copy or move the plug-in. Individual plug-in control parameters can be modified in real-time through provided dialog windows and maintained as named presets for re-usability.

4.9.2. LADSPA

LADSPA [9] has been the Linux audio plug-in standard for many years. There are literally hundreds of LADSPA plug-ins available for Linux. LADSPA plug-ins give the user many standard options such as Equalization, Filtering, Reverb, Chorus, Amp and speaker simulation, etc. Qtractor must be told the location (path in the file-system hierarchy) of the LADSPA plug-ins that have been installed by the user. This path can be specified in the **View / Options...** window, under the Plugins tab (see Illustration 4.13 below).

4.9.3. DSSI

DSSI plug-in support is available for DSSI effects plug-ins. DSSI Instrument plug-ins are not yet supported. You must have the core DSSI subsystem installed in order for this type of plug-in 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 **View / Options...** window under the Plugins tab, or with an environment variable.



*Illustration 4.13: The locations (paths) of LADSPA and DSSI audio plugins can be specified in the **View** -> **Options** window, under the **Plugins** tab.*

4.9.4. VST (Linux Native)

Native Linux VST plug-in 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.

Please see section 2.4.2 for complete information on building Qtractor with native VST support.

Note: Native Linux VST support does NOT include running of Windows® VST plug-ins. Please use the DSSI-VST wrapper when attempting to use this type of plug-in, and make sure your Windows® VST plug-ins are located within your DSSI path environment variable (the variable name is **VST_PATH**).

4.10. MIDI Instruments

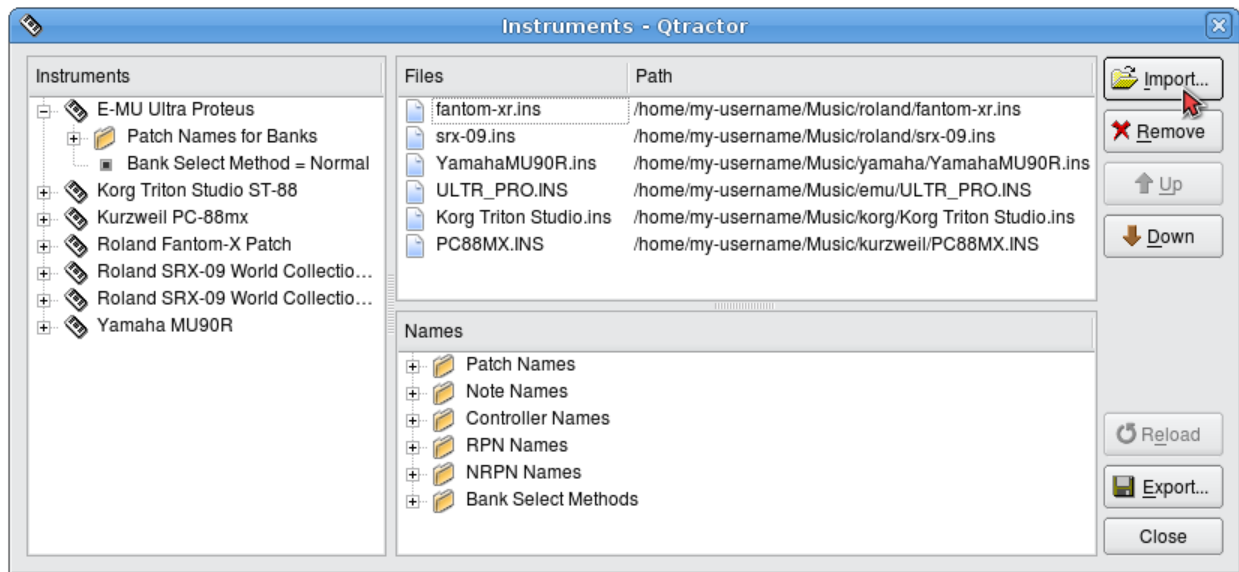


Illustration 4.14: Instruments window, where MIDI instrument definition files, for various popular MIDI tone generators, can be imported, exported and moved.

There are many MIDI hardware tone generators available from a variety of manufacturers such as Yamaha, Roland and Korg.. In Qtractor, information about the sound patches and sound banks of these tone generators is obtained from “instrument definition” files. Qtractor supports the instrument definition files used by the Cakewalk / Sonar [11] MIDI sequencer software, offering a convenient MIDI bank-select/program-change mapping for existing MIDI instrument patch names, and easier, intelligible selection of MIDI track channels. These Cakewalk / Sonar instrument definition files (.ins files) can be imported using the **View / Instruments...** window.

Cakewalk / Sonar instrument definition files for many popular MIDI tone generators and synthesizers can be downloaded from <http://www.cakewalknet.com/> in the “Downloads” section of the website.

4.11. MIDI Editor

Each MIDI clip content may be readily edited under a dedicated and fairly complete “piano-roll” type editor,, 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.

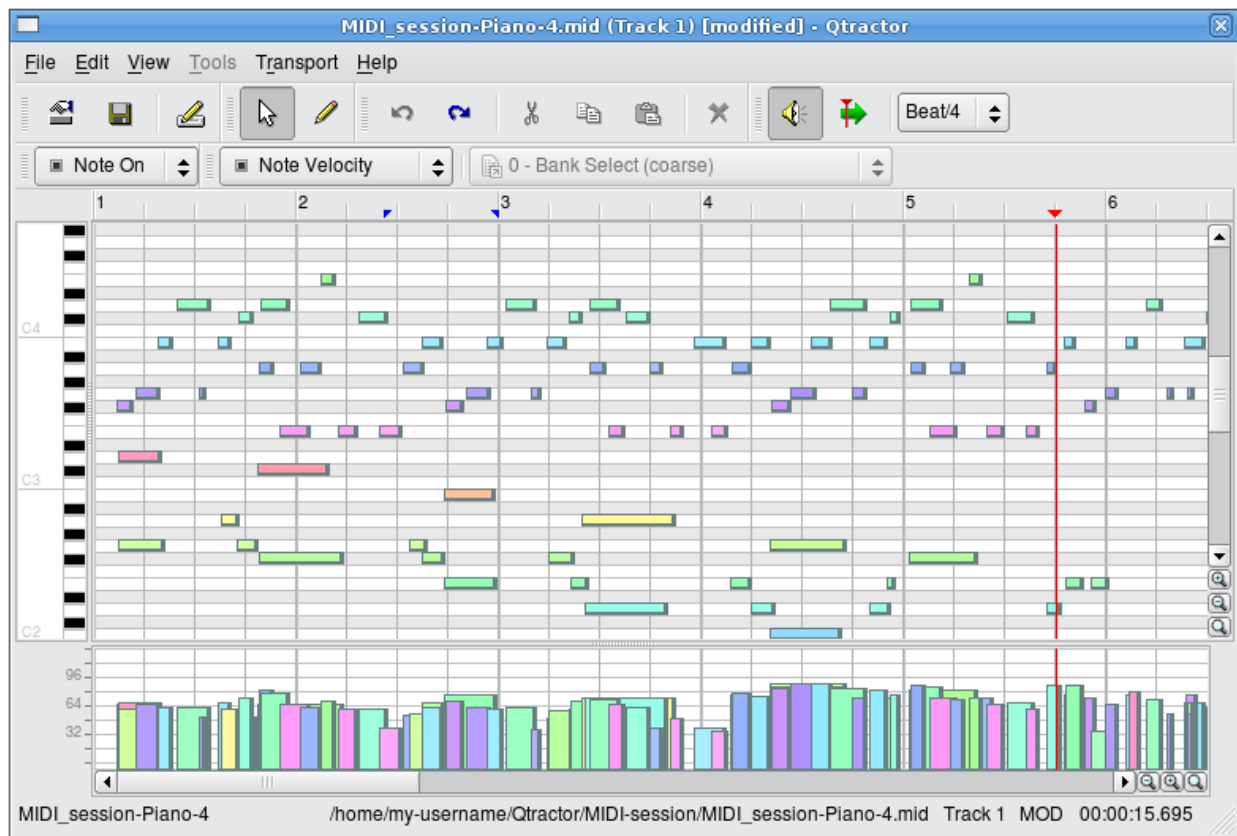


Illustration 4.15: MIDI Editor window, showing notes in a familiar “piano roll” type of graphical display, above, and a graph of their velocities below. The notes and their velocities can be selected and edited in many ways, using the mouse, menu commands and context menus.

Special 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 (all tracks reduced to one track, preserving channel assignment data) or format 1 (multi-track). The format for the SMF files (format 0 or 1) may be set in **View / Options...** under the MIDI tab.

4.12. Audio / MIDI Export

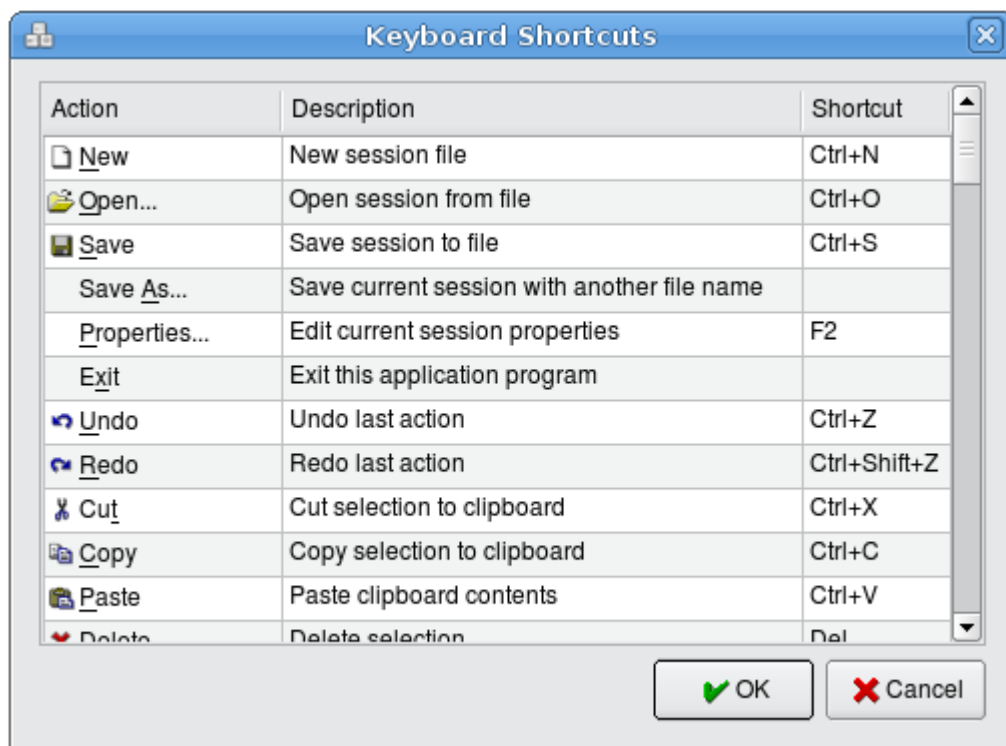


*Illustration 4.16: Audio export window, where the audio tracks in a session can be named and exported as a single audio track in the particular format previously specified by the user in the **View -> Options** window.*

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). The user-preferred format for exported audio files (OGG, FLAC, WAV, etc.) and MIDI files (SMF 0, all tracks reduced to single track or SMF 1, multi-track) can be set in the Options window (**View / Options...**).

4.13. 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.



*Illustration 4.17: Keyboard Shortcuts Editor window, available from the **Help / Shortcuts...** menu item*

Keyboard shortcuts may be customized to your preference by using the shortcuts editor. This editor may be found in the Help menu (**Help / Shortcuts...**).

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.

(continues on [qtractor 0.3.0 manual pt3](#))

qtractor 0.3.0 manual pt3

(continued from [qtractor 0.3.0 manual pt2](#))

5. Qtractor Main Menu

5.1. 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.

5.2. Edit Menu

- **Undo** – undo the last action.
- **Redo** – 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
- **Clip** – New..., Edit..., Split, Normalize and Export...

5.3. Track Menu

- **Add Track...** – add either a new audio or MIDI track.
- **Remove Track** – removes a track.
- **Track Properties...** – calls the track properties dialog.
- **Inputs** – shows current track inputs.

- **Outputs** – shows current track outputs.
- **State** – current track state: Record, Mute, Solo, Monitor.
- **Navigate** – current track navigation: First, Previous, Next, Last.
- **Move** – move current track navigation: Top, Up, Down, Bottom.
- **Auto-monitor** – current track auto-monitoring.
- **Import Tracks** – import either audio or MIDI track.
- **Export Tracks** – export either audio or MIDI track.

5.4. View Menu

- **Menu Bar** – toggle whether the menu bar is shown.
- **Status Bar** – toggle whether the Status Bar is shown.
- **Tool Bars** – toggles various tool bars on and off.
- **Files** – toggles whether the Files directory is on or off.
- **Messages** – toggles whether the Messages dialog is on or off.
- **Connections** – displays the Connections dialog.
- **Mixer** – displays the Mixer window.
- **Zoom** – zoom the main view: In, Out, Reset.
- **Snap** – change the snap-per-beat setting.
- **Refresh** – refresh the view contents.
- **Instruments...** – displays the instruments dialog.
- **Buses...** – opens the buses editor dialog.
- **Options...** – opens the program options dialog.

5.5. Transport Menu

- **Backward** – playhead jumps backward to beginning of range markers or session
- **Rewind** – playhead moves backward at greater than usual speed
- **Fast Forward** – playhead moves forward at greater than usual speed
- **Forward** – playhead jumps forward to marker or end of session
- **Loop** – play the range marked as a loop, over and over again
- **Loop Set** – mark selected range as a loop

- **Play** – playhead moves forward at normal speed
- **Record** – prepare to record audio or MIDI on record-enabled tracks
- **Punch** – record the range marked to punch in/out
- **Punch Set** – mark selected range to record punch in/out
- **Metronome** – turn the metronome’s audio or MIDI output on or off
- **Follow Playhead** – the tracks will scroll so that the playhead is always visible
- **Auto Backward** – playhead returns to beginning of range markers or session automatically when playhead is stopped
- **Continue Past End** – playhead will continue to move forward even after end of recorded tracks is reached

5.6. Help Menu

- **Shortcuts...** – lists the keyboard key combination equivalents of menu items and other commands
- **About...** – information about Qtractor
- **About Qt...** – information about Qt, the graphics toolkit used for Qtractor’s user interface.

6. Appendixes

6.1. References

Certain words, terms or items in the manual have a bracketed number after them, indicating a reference to items in this appendix, where sources are given for more information about those items.

[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** - The personal web presence of Rui Nuno Capela, creator and developer of Qtractor and other applications or application GUIs such as Qjackctl a GUI for jackd, Qsynth a GUI for fluidsynth, Qsampler, etc.

<http://www.rncbc.org/>

6.2 Colophon

This manual was produced using the free, open-source office suite OpenOffice.org office, and typeset using the free Liberation family of typefaces from Red Hat Linux which consist of the typefaces (fonts) Liberation Serif, Liberation Sans and Liberation Mono, all of which have the same metrics (proportions) as the proprietary Times New Roman, Arial and Courier New, respectively.

OpenOffice.org

<http://www.openoffice.org/>

Red Hats Liberation Typefaces

<https://www.redhat.com/promo/fonts/>

6.3 Contact Us

For questions, comments and suggestions regarding Qtractor, please contact:

- Rui Nuno Capela, rncbc@rncbc.org

For questions, comments and suggestions regarding the Qtractor manual, please contact:

- James Laco Hines, jhines@cebridge.net
- Stephen Doonan, stephen.doonan@gmail.com

Index

Aiff 18, 27

ALSA 4, 5, 9, 14, 15, 22, 30

Amp 24

Au 18, 27

Audio

- Clip properties & settings 19
- Clips 19
- Connections 22
- Exporting 27
- File types supported 18
- Files, viewing 18
- Importing audio files 18
- Input, getting audio data into Qtractor 9
- Plugins 23
- Plugins, DSSI 24
- Plugins, LADSPA 24
- Plugins, VST 6, 24
- Recording 20
- Sample rate 16

Audio export 27

Autoconf 5

Bus

- Buses in Mixer window 22
- Definition & general information 13

Cakewalk 25, 30

Cakewalk .ins (instrument definition) files 25

Chorus 24

Clip objects 19

Clip Objects 16

Clips

- Audio & MIDI 19
- Audio & MIDI, importing 19
- Audio properties & settings 19
- Definition & information 19
- Editing 20
- Recording 20

Compiling Qtractor

- Standard compilation 6
- With debugging code activated 7
- With VST support 6

Configuration 8

Connections 9, 16, 22

Connections

- Audio & MIDI 9

Contact us 31

Controller-10 22

Controller-7 22

Copy 26

Core dumps 7

Cubic 19

Cut 26

CVS 6

Data flow in, through & out of Qtractor 14

DBfs 22

Debian 5

Debugging 7

Downloading Qtractor

- Standard released source code 6
- “Bleeding edge” active-development source code 6

Drag-and-drop 26

DSSI 6, 23, 24

DSSI-VST 24

Edit menu, main workspace 28

Editing 20

Equalization 24

Exporting Audio or MIDI 27

Fade-in 19

Fade-out 19

Fedora 5

File menu, main workspace 28

Files 18

Files

- Importing audio & MIDI files 18
- MIDI instrument definition file 25
- Viewing session audio and MIDI files 18

Filtering 24

Flac 18, 27

Fluidsynth 30

Freewheel 27

G++ 5

Gain 19

GNU C++ 5

GPL 4

Graphical user interface, Qtractor's 15

GUI 4, 15

Help menu, main workspace 29

Importing audio & MIDI files 18

Input

- Audio and MIDI 9
- Installing Qtractor
- Compiling from source code 5
- Mandatory & optional prerequisites 5
- With optional functionality 5

Instruments

- Instrument definition file 25
- MIDI: software & hardware tone generators 25

JACK 4, 5, 10, 14, 15, 16, 22, 27, 30

Jackd 10, 30

Keyboard shortcuts for menu & other commands 27

LADSPA 5, 22, 23, 24, 30

Liberation typefaces, free typefaces from Red Hat 31

Liblo 6

Libmad 5, 18, 30

Librubberband 6

Libsamplerate 5, 16, 30

Libsndfile 18, 30

Libvorbis 5, 18, 30

Linear 19

Logarithmic 19

Main window & workspace 15

Menus

- Main menu items / commands 28

MIDI

- Cakewalk & Sonar MIDI instrument definition files 25
- Clips 19
- Connections 22
- Editing 26
- Editing, MIDI clips in main workspace 20
- Editor window 26
- Exporting 27
- Files, viewing 18
- Importing MIDI files 18
- Input, getting MIDI data into Qtractor 9
- Instrument definition file 25
- Recording 20
- Saving a track as an SMF file (Standard MIDI file) 26

MIDI Editor

- Description 26
- Editing MIDI pitch and velocity data 26
- Saving a MIDI track as an SMF file (Standard MIDI file) 26
- “Batch” editing functions for many notes or entire clips 26

MIDI export 27

Mix-down 22, 27

Mixer 16

Mixer

Buses 22

Mixer window 22

Plugins 22

Tracks 22

Monitoring 22

Move 26

Mp3 18

Normalize 26

Ogg 18, 27

Omni 11

OpenOffice & OpenOffice.org 31

Options 17

Pan 22

Paste 26

Piano-roll 21, 26

Plugins

- About LADSPA, DSSI & other audio plugins 23

Qjackctl 30

QjackCtl 22, 30

Qsampler 30

Qsynth 30

Qt 4 5, 30

Quantize 26

Randomize 26

Red Hat 5, 31

Redo 21

References to software & websites 30

Resize 26

Resizing 21

Reverb 24

RMS 21

Rncbc.org 30

Routing

- Definition & general information 13
- Methods of setting & changing 14
- Signal/data flow into, through & out of Qtractor 14
- Technical notes 14

Sample rate 16

Session, a Qtractor recording / Editing project

- Audio sample rate 16
- Files recorded or imported into a session 18
- Options: audio, MIDI, display, plugins, etc. 17
- Properties, basic session-wide settings 16
- Understanding a session 16

Settings (Qtractor settings file) 8

Shortcuts for menu & other commands 27

Signal flow in, through & out of Qtractor 14

SMF 18, 20, 26, 27

Snapping 21

Sonar 25

Sonar .ins (instrument definition) files 25

SoundTouch 30

Square 19

SUSE 5

Tempo 16

Time signature 16

Time stretch 19

Track

- About, audio or MIDI clips 21
- Colors 21

- Definition & general information 13
- Tracks area of main workspace 21
- Tracks in Mixer window 22

Track menu, main workspace 28

Tracker type of application 5

Tracks 21

Transport menu, main workspace 29

Transpose 26

Ubuntu 5

Undo 21

Velocity 19

View menu, main workspace 28

Volume 19, 22

VST 6, 23, 24

VST_PATH 24

VST-SDK 6

Wav 18, 27

Waveform 21

Windows

- Connections: routing, input & output 9, 22
- Main workspace 15
- Main workspace, tracks area 21
- MIDI Editor 26
- Mixer 22

WINE 24

Zooming 21

qtractor 0.5.x Part 0. Installing Qtractor

Part 0. Installing Qtractor

The different ways to install Qtractor and ensure your audio production environment is ready to make serious sounds.

Chapter 1, Introduction to Qtractor

Chapter 2, Installation

Chapter 3, Post-Installation Optimizations

Chapter 1. Introduction to Qtractor

Qtractor is a full-featured Digital Audio Workstation (DAW); a multi-track audio and MIDI recorder, editor, and mixer. Qtractor is free open-source software, licensed under the GPL, being developed by renowned programmer Rui Nuno Capela.

The functionality of **Qtractor** is contained within a graphical desktop environment that will be familiar to users of other popular multi-track recording/editing applications on any computer operating system, and follows the same design principles with many of the same or similar elements.

In addition to recording digital audio and MIDI, **Qtractor** provides an environment for multi-track clip-oriented composing techniques common in modern music-making and aims to be intuitive and easy to use, yet powerful enough for the serious recording enthusiast.

Note

Qtractor is not what is known as a “tracker” type of audio/MIDI application, although it has the potential to function in that way if needed. When used merely as an audio and/or MIDI recorder (a MIDI recorder was historically called a “sequencer”) or arranger, **Qtractor** is non-destructive, which means that the underlying files that contain the audio or MIDI data are not altered when those files are apparently cut into pieces, duplicated, pulled or pasted into a different order in time, or manipulated in any number of ways within the main Window (GUI interface) of **Qtractor**. However, when used as an audio or MIDI recorder, for example, or when editing previously recorded MIDI data in the dedicated MIDI editor, Qtractor’s actions can be destructive in the sense that newly recorded data (or altered MIDI data) replaces previously recorded data on the same track.

1.1. System Requirements

Qtractor’s target platform is **Linux**, using the **ALSA** (Advanced Linux Sound Architecture) and **JACK** (the Jack Audio Connection Kit) as the supporting infrastructure for recognizing sources of digital audio and MIDI (Musical Instrument Digital Interface) data, communicating with those sources and routing the data to and from various locations and programs (applications, including Qtractor) both inside and outside the computer and involving both software and hardware interfaces.

Suggestions of technical requirements for **Qtractor** are:

- PC running a distribution of the GNU+Linux operating system with a Kernel of 2.6.38.4 or higher for best realtime performance (older kernels require realtime patches from your distribution or from kernel.org)
- CPU capable of running reasonably modern software (the better your CPU, the better performance you will see in realtime audio effects and synthesis)
- A reasonable amount of RAM (an open **Qtractor** project requires RAM; the more you have, the more complex your projects can be and the faster they will respond)
- A sound card; the basic sound card that shipped with your computer is perfectly acceptable, although if you require simultaneous recording of multiple tracks, a card or audio interface with distinct inputs will be required
- Ample harddrive space for samples and audio files used in your projects.
- Attitude. Not required, but if you plan on being a rock star, then you should try to develop an rebellious attitude, and resolutely reject mainstream software

1.2. Get Involved

The **Qtractor** project welcomes all collaboration and review from the **Linux** audio developer and user community in particular, and the public in general.

See qtractor.sourceforge.net for contact information.

The docbook sources for this user manual are available via **git** from gitorious.org/slackermidia

Chapter 2. Installation

Installing **Qtractor** can be done in two ways: from your distribution's software repository, or by building from source code. For most users, installing from your distribution's repositories is the right choice, as it offers the easiest install and ensures timely and reliable updates.

2.1. Easy Install

To install software from your distribution's repository, open the software installer for your distribution (often called **Add/Remove Software** or sometimes a **Software Store**).

Search for Qtractor, and mark it for installation and proceed; there is no need to install additional packages since the installer will automatically install any software required for **Qtractor** to run.

2.2. Expert Install

If you absolutely require the latest features in a just-released or not-yet released version of **Qtractor**, or you simply prefer to build your programs manually, you may build **Qtractor** from source code.

In order to build from source, you must have a build environment, and the essential dependencies that **Qtractor** requires. Depending on your distribution, the names of the installable build tools will vary, but the software components themselves are:

- **g++** - the GNU C++ Compiler
- **gcc** - the GNU C Compiler; not strictly required in this case, but good to have in a build environment, and in some distributions this may bring in other useful dependencies
- **Autoconf** -
- **Automake** - the GNU Makefile generator
- **Qt4** (core,gui,xml) - C++ class library and tools for cross-platform development and internationalization, available from your distribution's software repository (be sure to install all "development" components) or directly from trolltech.org/products/qt
- **libsndfile** - C library for reading and writing files containing sampled sound, available from your distribution (be sure to install all "development" components) or directly from mega-nerd.com/libsndfile
- **LADSPA** - Linux Audio Developer's Simple Plugin API; available from you distribution's repository (be sure to install all "development" components) or from ladspa.org

Additional support libraries may be installed to enhance the abilities of **Qtractor**:

- **libvorbis** - the Ogg Vorbis encoder and decoder by Xiph
- **libmad** - high-quality MPEG audio decoder, available from your repository or from underbit.com/products/mad
- **libsamplerate** - the secret rabbit code, a C library for audio sample rate conversion, available from your distribution's repository or from mega-nerd.com/libsndfile
- **librubberband** - the Rubber Band Audio Time Stretcher, an audio time-stretching and pitch-shifting library, available from your repository or breakfastquay.com/rubberband
- **liblo** - Lightweight OSC implementation (needed for DSSI GUI support), available from your repository or liblo.sourceforge.net
- **DSSI** - an API for soft synth plugins with custom user interfaces dssi.sourceforge.net
- **VST-SDK** - Steinberg's Virtual Studio Technology; this SDK permits the use of Linux-native plugins with Qtractor. To install VST-SDK:

1. Establish a developer account at steinberg.net/en/company/3rd_party_developer.html, confirm the registration, and log in
2. Download VST 2.4
3. Unzip the source directory and copy the files to the appropriate directory; usually /usr/local/include

```
su -c 'cp vstsdk2.4/plugininterfaces/vst2.x/aeffect* /usr/local/
include/'
```

2.2.1. Download the Qtractor Code

Stable releases of **Qtractor** can be downloaded directly from qtractor.sourceforge.net/qtractor-index.html#Downloads

Note

If you intend to develop **Qtractor** then you will want to run “bleeding edge” code. This is not recommended for a production system and should be used only for development purposes. Check out the source code from its CVS repository via anonymous (pserver) access:

1. At the command line, checkout the code via SVN:

```
svn co https://qtractor.svn.sourceforge.net/svnroot/qtractor/trunk
qtractor-svn
```

2. Prepare the configure script on the newly created qtractor source tree directory:

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

3. Build and install.

2.2.2. Compiling and Installing Qtractor

After downloading Qtractor, decompress and extract the archive:

```
tar -xf qtractor-0.x.x.tar.gz
```

Change directory into the resulting **qtractor** directory. Once inside the **qtractor** directory, the usual building commands apply:

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

To see all configuration options before entering the command sequence above, type **./configure --help**

After typing the **configure** and **make** commands and waiting until the program has finished being compiled, become an administrator of your system (using either the **sudo** or **su** command to become, temporarily, the **root** user) and finish the installation by entering:

```
make install
```

The executable binary code (in otherwords, the **Qtractor** application itself) and associated desktop and icon files are copied to common, standard system locations.

Chapter 3. Post-Installation Optimizations

There are some considerations after you have installed **Qtractor** which may help you optimize your system for its best possible audio performance.

Previous to the Linux Kernel 2.6.38.4, a realtime kernel was a must-have component in audio production. Since 2.6.38, new options have developed to provide a seamless audio production experience, depending largely on what your needs are from your system.

If you are not sure whether you'll need realtime, it is safe to assume that you will need either:

- A 2.6.38.4 or greater kernel regardless of your expected workload
- A realtime kernel if a 2.6.38.4 or better kernel is not conveniently available from your distribution
- An optimized realtime kernel if you anticipate live recording and multitracking

3.1. Real-Time Kernel

To discover what kernel your computer is currently running, look in KInfoCenter (in KDE4) or System Information (in Gnome3).



Or simply open a terminal:

```
uname -r
```

If your computer is running a kernel previous to 2.6.38.4, then you should either update your kernel, or install a realtime version of your kernel.

To upgrade the Linux kernel, use your distribution's repository, searching for the terms **realtime** or **rt**.

If your distribution does not have repositories, or realtime kernels, then you can of course download the source code and compile the kernel yourself. The same guiding principles apply; as long as you are building 2.6.38.4 or above, then a simple **make oldconfig** will render a modestly pre-emptive kernel suitable for near-realtime use. For extreme low-latency, apply realtime patches, available from kernel.org/pub/linux/kernel/projects/rt

3.2. QJackCtl

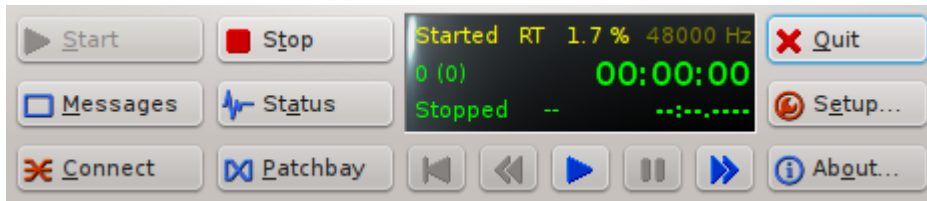
In order for **Qtractor** to run, **JACK** must be running in the background. **JACK** is a kind of patchbay for Linux (technically a "sound server"), enabling you to route sound in and out of **Qtractor** and other sound applications.

Some distributions may configure **Qtractor** such that **JACK** starts automatically when **Qtractor** is launched; others may not. If you launch **Qtractor** from your application menu and receive

JACK errors in the message log at the bottom of the **Qtractor** window, then quit **Qtractor** and start **JACK** manually.

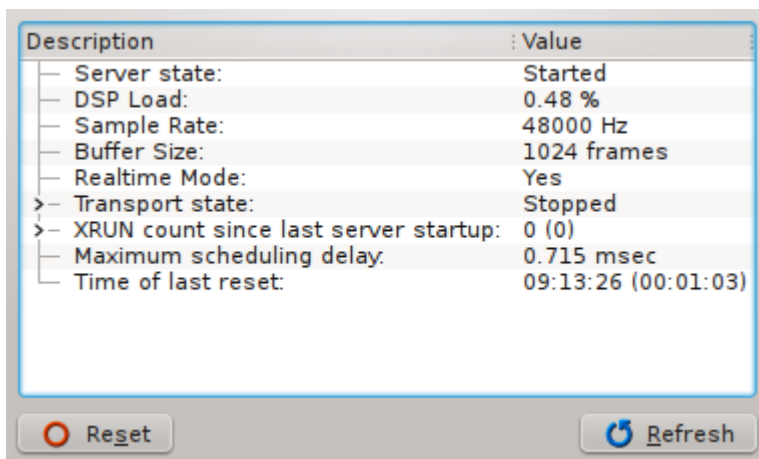
The easiest way to start **JACK** is with **QJackCtl** (“Q JACK Control”), a control panel and global, synchronized timecode display. If this is not installed, install it from your distribution’s repository or from qjackctl.sourceforge.net

If necessary, launch **QJackCtl** and click the **Start** button.



This should instate realtime privileges and begin a global, synchronized timecode clock that will allow **Qtractor** to use all available inputs and outputs just like a mixing board would in the studio.

To view the status of **JACK**, click the **Status** button and review the resulting report. **JACK** should be reported as running, in realtime mode.



3.2.1. Troubleshooting

If you are experiencing unexpected results with **QJackCtl**, then here are some things to check:

- **JACK** does not indicate realtime mode in the Status window, on a system using the **PAM** security model.
 1. Click the **Setup** button to open preferences. Activate **Realtime** in the left column, and then quit **QJackCtl** while you configure your system further.
 2. Make sure you have a realtime-capable (ie, 2.6.38.4 or greater, or an RT-optimized kernel of any variety) kernel installed and that applications have permission to utilize the realtime mode. On distributions using **PAM**, look in `/etc/security/limits.conf` and ensure that there is a user group being granted realtime (or **rtprio** in **PAM** terminology) privileges. If **rtprio** is being granted to a group, then you should make sure that you are in that group. For instance, if the group **audio** is being granted realtime privileges then execute `cat /etc/group | grep audio` and look to see that your userinput is listed as a member of the audio group. If not, execute

usermod -a -G audio youruserinput as root to add your user to the audio group.
Warning If **limits.conf** does not exist, then your system does not use **PAM**. If no group is given realtime priority, create a group and add yourself to the group. This must be done with root privileges: **groupadd realtime usermod -a -G realtime yourUserName** Then add realtime priority permissions to the group (and accordingly any user in that group): **@realtime hard rtprio 20 @realtime soft rtprio 10**

- **JACK** does not indicate realtime mode in the Status window, on a system not using the **PAM** security model.

1. Click the **Setup** button to open preferences. Activate **Realtime** in the left column, and then restart **QJackCtl** while you configure your system further.
2. To enable realtime priority for your user applications, install **set_rlimits** from your repository or directly from physics.adelaide.edu.au/~jwoithe
3. As root, open **/etc/set_rlimits.conf.new** in your favourite text editor. Add two lines: **@audio /usr/local/bin/jackd nice=-1 rtprio=80 memlock=100000 @audio /usr/local/bin/qtractor nice=-1 rtprio=80 memlock=100000** Save the file as **/etc/set_rlimits.conf**
4. Create a custom launcher so that when you start your realtime applications, they are started in the syntax **set_rlimits qtractor**. In KDE, do this by right-clicking on the **K Menu** and edit the launch commands for **QJackCtl** and **Qtractor**. In Gnome, edit as root the **.desktop** file for the applications (ie, **/usr/share/applications/qtractor.desktop**) such that the **Exec=** line executes **set_rlimits qtractor**

- **JACK** does not start when **Qtractor** is launched.

1. Your distribution may not have configured **JACK** to automatically start upon **Qtractor**'s launch.
2. The easiest workaround for this is to launch **QJackCtl** manually first, and then launch **Qtractor**.

qtractor 0.5.x Part 1. Quick Start

Part 1. Quick Start

Diving into Qtractor; a quick start guide for the basic workflow.

Chapter 4, Getting Sound Into Qtractor

Chapter 5, Effects and Filters

Chapter 6, Automation in Qtractor

Chapter 7, Bouncing the Project

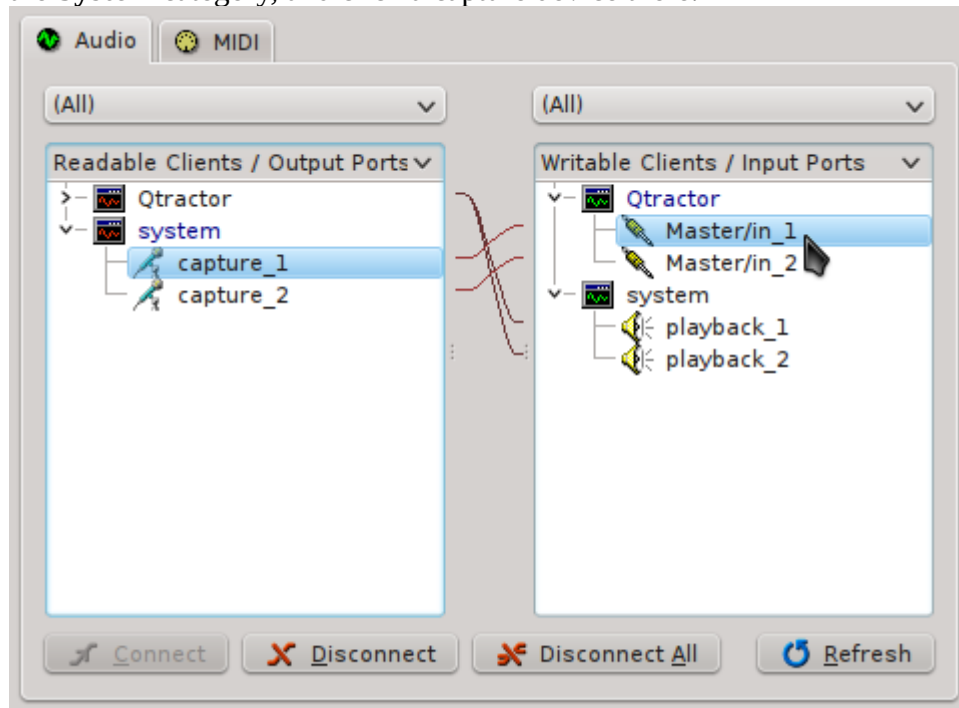
Chapter 4. Getting Sound Into Qtractor

Getting started with basic music production in Qtractor is simple as long as all of the requirements are installed. Review the Installation chapters to ensure you've installed all the necessary components before beginning. As long as you've done that, you'll find Qtractor a robust digital production environment with plenty of knobs and dials to tweak.

1. Start the **QJackCtl** application (recommended), or start **jackd** from a shell. Once launched, press the **Start** button to start JACK.
2. Launch **Qtractor** from your applications menu, dock, or launcher.
3. It's good practise to start every session by saving. It might seem strange to save an as yet empty session, but it's better to save an empty session than to start creating your masterpiece and have data files and MIDI files scattered all throughout your hard drive. Saving first is a good way to instantiate an environment in which you can keep all of your files and sounds organized and consolidated. To save, click the **File** menu and select **Save As**. In the **Name** field, name your session. For the **Directory** field, click the **Directory** icon to create a new, empty directory for your session files and click **OK**. Save your session by clicking the **OK** button; a **Save Session** dialogue will open so that you can now navigate to the directory you've created and name your session file, which will appear as a **.qtr** file. Click the **Save** button to confirm.
4. Now you're ready to start producing! Obviously there are different means to creating music in **Qtractor**, but nearly each one would create with creating a new track. To create an empty track in your **Qtractor** workspace, click on the **Track** menu and select **Add Track**. In the **Track** dialogue, name your track in the **Name** field. For **Type**, choose between Audio or MIDI, depending on what kind of data you wish to use in the track. If you have a specific set of inputs and outputs that you wish to use for the track, you can set that here, or you can leave it on the default and manipulate it later as needed. Likewise, if you know the MIDI bank or patch selections, you can set it here, or leave it as default values and configure it later. Click **OK** to create the track.

5. Now that you have an empty track, you should put some sound into it. You can do this in four ways:

- **Record audio directly into the track** - to record directly into **Qtractor**, you must define your input device in your central patchbay (ie, **QJackCtl**). To do this, choose **View** menu > Windows > Connections. In the left column, choose the source of sound (a device that, from **Qtractor**'s perspective, is outputting sound, hence the "output" label). Your microphone exists as part of your computer system, so choose the **System** category, and click a capture device there.



Now choose the sound source's destination in the right column. You are routing the sound from the microphone into **Qtractor**, so choose the **Qtractor** category and select the corresponding input port, such as Master/in_1. Click the **Connect** button if they are not already connected. Repeat this process for the right channel, and then close the window. In the track label on the left of the **Qtractor** window, click the **R** button to arm the track for recording. Click the **Record** button in the top menu bar to activate Recording mode. Click the **Play** button in the top menu bar to begin.

* ***Insert an existing audio file into the track*** - if you're using pre-recorded material from a live performance, a sound booth session, or a sample or loop collection, then there is no need to set up a recording source such as a microphone or line-in. You'll simply add files to your **Qtractor** session and drag them into your workspace.

To add sounds files to your session, go to the **Clip** menu and select **Import**. Choose the file you wish to import, and it will be added into your empty track. Notice that it is also added in the **Files** panel on the right of the **Qtractor** window; so to add files without adding them to a track, you can right-click in the **Files** panel and choose **Add Files**.

* ***Play MIDI into the track*** - similar to recording audio directly into **Qtractor**, you can connect a MIDI controller to your computer and record MIDI data into a track. The MIDI data will be used to trigger a sound source; the simplest and most direct method is to use the MIDI to trigger a software synthesizer, but of course the MIDI signal could also be routed back out of the computer into a hardware synth as well.

Assuming you are using a USB-based MIDI controller such as the Oxygen or Axiom series, or AKAI LPK series, and so on, then you must first define the source of your MIDI signal. Click the **View** menu > Windows > Connections to see your central patchbay (ie, QJackCtl). In the left column, choose the source of your MIDI signal (a device that, from **Qtractor**'s perspective, is outputting the signal, hence the "output" label). Your USB controller should be listed as a source, so open its listing and click the MIDI channel listed there. Now choose the MIDI signal's destination in the right column. You are routing the sound from your hardware controller to **Qtractor**, so choose the **Qtractor** category and select the corresponding input port, such as 0:Master Click the **Connect** button if they are not already connected. Close the window. If you start playing now, you'll be sending MIDI data without hearing any feedback whatsoever, because you have not yet configured the MIDI signal to trigger sound. So that you can hear what you're playing, you must insert a plugin synthesizer into the track. With your destination track selected, click on the **Track** menu and select **Track Properties**. In the **Track** dialogue, click the **Plugins** tab. In the **Plugins** tab, click the **Add** button on the right to see a list of available software plugins. In the **Plugins** dialogue, select the type of plugin you wish to use with the top right button. Software synths on Linux are usually of the DSSI variety; from the list of DSSI synths, select the one you want to play and click the **Activate** checkbox in the lower left corner. Click the **OK** button in the bottom right corner of the window to proceed. **Note** If you don't have any plugins installed, you can install them from your distribution or directly from their project websites. One of the best soft synths for immediate gratification is **whySynth**, which bundles some nice, ready-to-use patches. You will need to restart Qtractor in order for your new soft synths to be available. Back in the **Track** dialogue, select the **Track** tab again. In the MIDI/Instrument panel, choose an instrument from the top dropdown menu; using **WhySynth** as an example, you would select **WhySynth_20100922 DSSI plugin**. Choose a **Bank** and **Program**. If you're not familiar with **WhySynth**, you can select any of its three Banks and any of the Programs contained in them. You should be able to instantly audition the sounds on your USB controller. When you've selected the sound you'd like to use, click the **OK** button. In the track label on the left of the **Qtractor** window, click the **R** button to arm the track for recording. Click the **Record** button in the top menu bar to activate Recording mode. Click the **Play** button in the top menu bar to begin.

* ***Insert existing MIDI data into the track*** - if you want to use pre-made MIDI files then simply add the files to your **Qtractor** session.

To add MIDI files to your session, go to the **Track** menu and select **Import Tracks > MIDI**. Choose the file you wish to import, and it will be added into a new track. Notice that it is also added in the **Files** panel on the right of the **Qtractor** window; so to add files without immediately adding them to your workspace, right-click in the **MIDI** tab of the **Files** panel and choose **Add Files**. **Note** The MIDI file you import may or may not have generic MIDI instrument assignments. Either way, you can always adjust the MIDI instrument assigned to that track via the **Track Properties** dialogue, as you would for a track into which you're recording MIDI.

Chapter 5. Effects and Filters

You know how to create tracks and insert sound either in the form of audio files or MIDI data. In addition to putting sound into tracks, **Qtractor** can pipe tracks through filters and effects.

The most popular format for sound filters and effect units in Linux is LADSPA, providing the free equivalent to VST, RTAS, or AU. The two ubiquitous Linux plugin packs are the **Steve Harris LADSPA Plugins** and the **CALF Plugins**, which should be available from your repository or from their respective websites.

If you have not installed them yet, install them now and then re-launch **Qtractor** so that it will detect them as available filters.

Effects occur upon the sound itself, so they can be applied to Audio tracks as well as MIDI tracks. To place an effect on a track:

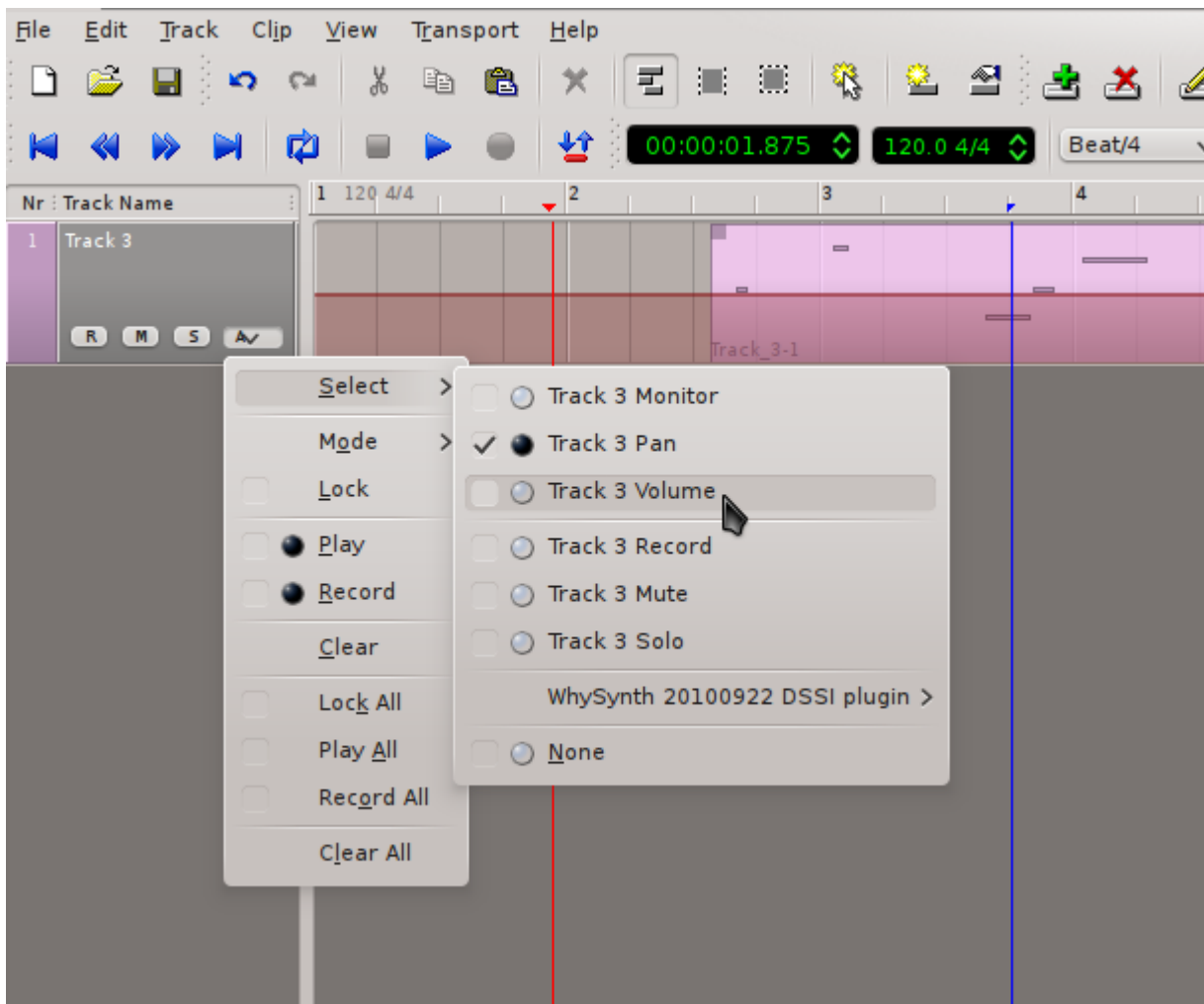
1. Select the track you want to pipe through an effect and select it by clicking the track label on the left of the **Qtractor** workspace.
2. Click on the **Track** menu and select **Track Properties**.
3. In the **Track** dialogue, click on the **Plugins** tab.
4. Click the **Add** button on the right of the window.
5. In the **Plugins** dialogue, choose the type of plugin you want to use with the filter button in the top right corner; more than likely, you will be using a LADSPA plugin.
6. A very common effect, of course, is reverb. Filter the choices from all the LADSPA plugins by typing “verb” into the search bar along the top. You may see one or two results, such as **GVerb** and **Plate Reverb**, so choose one of these. Place a tick in the checkbox in the lower left corner of the screen, labeled **Activate** and click **OK**.
7. Should you need to modify the reverb filter, right click on the plugin in the **Track** window and choose **Properties**. This will open the control panel for the effect, where you can change the attributes of the effect.

Chapter 6. Automation in Qtractor

Nearly every aspect of any track can be automated in **Qtractor**, from the basics like volume and panning, to the very minute like the LFO frequency of a soft synth, or the levels of an effect.

To work on the automation of your tracks:

1. Click the **automation** button in the track control of your track.



1. Choose from the popup menu the attribute you wish to automate.
2. Notice that an overlay appears over your track, representing the normal level of that attribute. To enter automation mode, click the **Edit** menu > **Select Mode** > **Automation**. In this mode, click the automation overlay to adjust levels.
3. To differentiate different attributes that you're automating, it can be helpful to modify the colour of the automation overlay. To do this, click the **automation** button in the track list and choose **Mode** > **Colour** and pick a new shade for that automation overlay. To add automation for another attribute, click the **automation** button again, and navigate to **Select** to choose the next attribute. The new automation overlay appears in the default colour, and you can now modify its levels.
4. Leave automation mode by toggling off **Edit** menu > **Select Mode** > **Automation**

Chapter 7. Bouncing the Project

Once you've finished your song, you obviously want to export the piece so that people can listen to it without having to have your **Qtractor** source files. This process is sometimes called "bouncing" a track, or exporting a "mixdown" or simply "exporting" the song.

Since it's possible to have multiple sources contributing to your final product (external hardware synths feeding sound into Qtractor while soft synths play over pre-recorded audio files to the beat of a Hydrogen drum machine on another virtual desktop, for example) it would make no sense to treat the exporting process in the same way as a wave form editor (such as **Audacity**) would.

Qtractor's bounce process is literally to play all the sound sources in sync whilst recording what is playing back into **Qtractor** itself.

Note

Different people deal with bouncing and exporting in different ways, depending on their preference, their creative needs, and the capabilities of their computers. For instance, you could bounce each MIDI track individually to an audio track and then export all audio tracks into one self-contained track. Or you can simply set **Qtractor** to record and then play everything back into one audio track, which you then export.

There is no right or wrong way to do it. Either do it all in one go at the end or do it step by step. Usually the raw power of your computer will be the deciding factor.

To perform the final export of your completed work:

1. Create a new Audio track: **Track > Add Track**
2. Label the track, ie, "Bounce"
3. Open the **Connections** window: **View > Windows > Connections**
4. In the **Connections** window, connect the Master/Out of **Qtractor** to the Master/In of **Qtractor**, such that the output of all sound managed by **Qtractor** is being directed to the input of **Qtractor** for recording. **Warning** Be sure to disable any capture device such as microphones or unused line-ins. You'll be recording all the live sound going into **Qtractor**, so you don't want accidental line noise, hums, or room tone.
5. Set the in and out points for your recording by clicking once on the top timeline at the point you want the recording to stop; a blue transport line will be anchored where you click. Scroll back to the beginning of your project and click again in the top timeline to mark the in point with an opening blue transport bar.
6. Click the **Punch In/Out** button in the top menu bar to limit playback between your markers. **Note** If you do not set in and out points, you will need to stop the recording manually.
7. Arm your *Bounce* track for recording. Click the **Record** button in the top menu bar. Make sure that your transport (playhead) is at the beginning of your track! When ready, press the **Play** button in the top menu bar. **Warning** Bouncing a track is a realtime process. Do not use your computer while you're bouncing a track!

When you've bounced your song to its own track, save the project! Solo the bounce track and listen to the recording for quality assurance, and then you are ready to export that track as a self-contained, distributable file.

To export a single track to disk:

1. Select the track or tracks you wish to export. If you bounced everything into one track, then it will just be that track, which you can solo using the **Solo** button on the track controls. If you are exporting a project consisting of only in-project audio, then leave them all activated and continue.
2. Choose **Track > Export Tracks > Audio**

3. In the **Export Audio** dialogue, choose a location and filename for your file. Choose the range of what will be exported; you can export the entire session, or manually set a Punch In/Out range, or specify the range in timecode, frames, or bars/beats/ticks. If you have multiple output sources, choose the appropriate source (probably your Master Output).
4. Click the **OK** button to begin the export.

Qtractor User Manual

Qtractor User Manual

Copyright © 2012 Seth Kenlon

Permission is granted to copy, distribute and/or modify this document under the terms of the [GNU Free Documentation License, Version 1.3](#) or any later version published by the Free Software Foundation; with no Invariant Sections, no Front-Cover Texts, and no Back-Cover Texts. A copy of the license is included in the section entitled Appendix A, GNU Free Documentation License.

A Guide for audio producers both new and experienced.

Introduction to Qtractor

Qtractor is a full-featured [Digital Audio Workstation](#) (DAW); a multi-track **audio** and **MIDI recorder, editor, and mixer**. Qtractor is free open-source software, licensed under the GPL, being developed by renowned programmer Rui Nuno Capela.

The functionality of Qtractor is contained within a graphical desktop environment that will be familiar to users of other popular multi-track recording / editing applications on any computer operating system, and follows the same design principles with many of the same or similar elements. In addition to recording digital audio and MIDI, Qtractor provides an environment for multi-track cliporiented composing techniques common in modern music-making and aims to be intuitive and easy to use, yet powerful enough for the serious recording enthusiast.

Note

Qtractor is not what is known as a “tracker” type of audio/MIDI application, although it has the potential to function in that way if needed.

When used merely as an audio and / or MIDI recorder (a MIDI recorder was historically called a “sequencer”) or arranger, Qtractor is **non-destructive**, which means that the underlying files that contain the audio or MIDI data are not altered when those files are apparently cut into pieces, duplicated, pulled or pasted into a different order in time, or manipulated in any number of ways within the main Window ([GUI](#) interface) of Qtractor.

However, when used as an audio or MIDI recorder, for example, or when editing previously recorded MIDI data in the dedicated MIDI editor, Qtractor’s actions can be destructive in the sense that newly recorded data (or altered MIDI data) replaces previously recorded data on the same track.

System Requirements

Qtractor’s target platform is [Linux](#), using the [ALSA](#) (Advanced Linux Sound Architecture) and [JACK](#) (the Jack Audio Connection Kit) as the supporting infrastructure for recognizing sources of [digital audio](#) and [MIDI](#) (Musical Instrument Digital Interface) data, communicating with those sources and routing the data to and from various locations and programs (applications, including

Qtractor) both inside and outside the computer and involving both software and hardware interfaces. Suggestions of technical requirements for Qtractor are:

- PC running a distribution of the GNU+Linux operating system with a Kernel of 2.6.38.4 or higher for best [Real-Time](#) performance (older kernels require realtime patches from your distribution or from [kernel.org](#)) - CPU capable of running reasonably modern software (the better your CPU, the better performance you will see in real time audio effects and synthesis)
- A reasonable amount of RAM (an open Qtractor project requires RAM ; the more you have, the more complex your projects can be and the faster they will respond)
- A sound card; the basic sound card that shipped with your computer is perfectly acceptable, although if you require simultaneous recording of multiple tracks, a card or audio interface with distinct inputs will be required
- Ample harddrive space for samples and audio files used in your projects.
- Attitude. Not required, but if you plan on being a rock star, then you should try to develop an rebellious attitude, and resolutely reject mainstream software

Dedication

[GNU/Linux](#) users often cite the “community” as one of the greatest appeals of using their OS. There must be a lot of truth to that, because this book would not exist without the collaboration, hours of troubleshooting, and tech support from everyone in the [freenode](#) channels #opensourcemusicians and #oggcastplanet and obviously without Qtractor this manual would not exist, so the greatest thanks goes to [Rui Nuno Capela](#), for the application that has enabled and inspired me and many other musicians all over the world to get those pesky tunes that pop into our heads down into actual sound waves that other people can hear, too.

Get Involved

The Qtractor project welcomes all collaboration and review from the [Linux audio developer and user community](#) in particular, and the public in general.

See [qtractor.sourceforge.net](#) for contact information.

The docbook sources for this user manual are available via git from [gitorious.org/slackermidia](#).

Installation

Installing Qtractor can be done in two ways : from your distribution’s software repository, or by building from source code. For most users, installing from your distribution’s repositories is the right choice, as it offers the easiest install and ensures timely and reliable updates.

Easy Install

To install software from your distribution’s repository, open the *software installer* for your distribution (often called *Add/Remove Software* or sometimes a *Software Store*). Search for Qtractor, and mark it for installation and proceed; there is no need to install additional packages since the installer will automatically install any software required for Qtractor to run.

Expert Install

If you absolutely require the latest features in a just-released or not-yet released version of Qtractor, or you simply prefer to build your programs manually, you may build Qtractor from source code. In order to build from source, you must have a build environment, and the essential dependencies that Qtractor requires. Depending on your distribution, the names of the installable build tools will vary, but the software components themselves are:

- [g++](#) - the GNU C++ Compiler
- [gcc](#) - the GNU C Compiler; not strictly required in this case, but good to have in a build environment, and in some distributions this may bring in other useful dependencies
- [Autoconf](#) - Automake - the GNU Makefile generator
- [Qt4](#) (core,gui,xml) - C++ class library and tools for cross-platform development and internationalization, available from your distribution's software repository (be sure to install all "development" components) or directly from [digia.com](#)
- [libsndfile](#) - C library for reading and writing files containing sampled sound, available from your distribution (be sure to install all "development" components) or directly from [mega-nerd.com/libsndfile](#)
- [LADSPA](#) - Linux Audio Developer's Simple Plugin API; available from you distribution's repository (be sure to install all "development" components) or from [ladspa.org](#) Additional support libraries may be installed to enhance the abilities of Qtractor:
 - [libvorbis](#) - the Ogg Vorbis encoder and decoder by Xiph
 - [libmad](#) - high-quality MPEG audio decoder, available from your repository or from [underbit.com/products/mad](#)
 - [libsamplerate](#) - the secret rabbit code, a C library for audio sample rate conversion, available from your distribution's repository or from [mega-nerd.com/libsndfile](#)
 - [librubberband](#) - the Rubber Band Audio Time Stretcher, an audio time-stretching and pitch-shifting library, available from your repository or [breakfastquay.com/rubberband](#)
 - [liblo](#) - Lightweight OSC implementation (needed for DSSI GUI support), available from your repository or [liblo.sourceforge.net](#)
 - [DSSI](#) - an API for soft synth plugins with custom user interfaces [dssi.sourceforge.net](#)
 - [VST-SDK](#) - Steinberg's Virtual Studio Technology; this SDK permits the use of Linux-native plugins with Qtractor. To install VST-SDK:
 - Establish a developer account at [Steinberg](#) ; confirm the registration, and log in
 - Download VST 2.4
 - Unzip the source directory and copy the files to the appropriate directory; usually `/usr/local/include`

```
su -c 'cp vstsdk2.4/plugininterfaces/vst2.x/aeffect*/usr/local/include/'
```

Download the Qtractor Code

Stable releases of Qtractor can be downloaded directly from qtractor.sourceforge.net.

Compiling and Installing Qtractor

If you intend to develop Qtractor then you will want to run “bleeding edge” code. This is not recommended for a production system and should be used only for development purposes ; check out the source code from its CVS repository via anonymous *pserver* access:

At the command line, checkout the code via [SVN](#) :

```
svn co https://qtractor.svn.sourceforge.net/svnroot/qtractor/trunk qtractor-svn
```

Prepare the configure script in the newly created `qtractor-svn` source tree directory :

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

Build and install.

After downloading Qtractor, decompress and extract the archive :

```
tar -xf qtractor-0.x.x.tar.gz
```

Change directory into the resulting qtractor directory. Once inside the qtractor directory, the usual building commands apply :

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

To see all configuration options before entering the command sequence above, type

```
./configure --help
```

After typing the configure and make commands and waiting until the program has finished being compiled, become an administrator of your system (using either the `sudo` or `su` command to become, temporarily, the root user) and finish the installation by entering:

```
make install
```

The executable binary code (in other words, the Qtractor application itself) and associated desktop and icon files are copied to common, standard system locations.

There are some considerations after you have installed Qtractor which may help you optimize your system for its best possible audio performance.

Previous to the Linux Kernel 2.6.38.4, a Real-Time kernel was a must-have component in audio production. Since 2.6.38, new options have developed to provide a seamless audio production experience, depending largely on what your needs are from your system.

If you are not sure whether you’ll need Real-Time, it is safe to assume that you will need either:

- A 2.6.38.4 or greater kernel regardless of your expected workload

- A realtime (or “low-latency”) kernel if a 2.6.38.4 or better kernel is not conveniently available from your distribution
- An optimized realtime kernel if you anticipate live recording and multitracking

Real-Time Kernel

To discover what kernel your computer is currently running, look in *KInfoCenter* (in KDE4) or *System Information* (in Gnome3).



Or simply open a terminal :

```
uname -r
```

If your computer is running a kernel previous to 2.6.38.4, then you should either update your kernel, or install a realtime version of your kernel.

To upgrade the Linux kernel, use your distribution’s repository, searching for the terms realtime or rt. If your distribution does not have repositories, or realtime kernels, then you can of course download the source code and compile the kernel yourself. The same guiding principles apply; as long as you are building 2.6.38.4 or above, then a simple make oldconfig will render a modestly pre-emptive kernel suitable for near-realtime use. For extreme low-latency, apply realtime patches, available from kernel.org/pub/linux/kernel/projects/rt.

QJackCtl

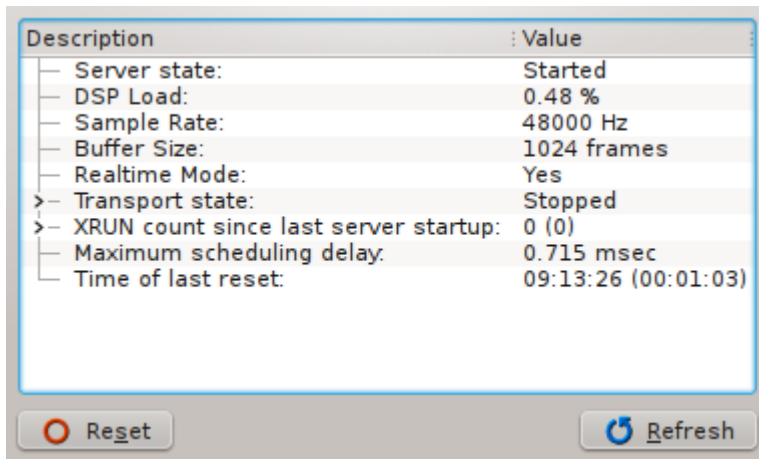
In order for Qtractor to run, [JACK](#) must be running in the background. JACK is a kind of patchbay for Linux (technically a “sound server”), enabling you to route sound in and out of Qtractor and other sound applications.

Some distributions may configure Qtractor such that JACK starts automatically when Qtractor is launched; others may not. If you launch Qtractor from your application menu and receive JACK errors in the message log at the bottom of the Qtractor window, then quit Qtractor and start JACK manually.

The easiest way to start JACK is with [QJackCtl](#) (“Q JACK Control”), a control panel and global, synchronized timecode display. If this is not installed, install it from your distribution’s repository or from qjackctl.sourceforge.net If necessary, launch QJackCtl and click the Start button.



This should instate realtime privileges and begin a global, synchronized timecode clock that will allow Qtractor to use all available inputs and outputs just like a mixing board would in the studio. To view the status of JACK, click the *Status* button and review the resulting report. JACK should be reported as running, in realtime mode.



Troubleshooting

If you are experiencing unexpected results with QJackCtl, then here are some things to check:

JACK does not indicate realtime mode in the Status window, on a system using the PAM security model.

- Click the *Setup* button to open preferences. Activate *Realtime* in the left column, and then quit QJackCtl while you configure your system further.
- Make sure you have a realtime-capable (ie, 2.6.38.4 or greater, or an RT-optimized kernel of any variety) kernel installed and that applications have permission to utilize the realtime mode. On distributions using PAM, look in `/etc/security/limits.conf` and ensure that there is a user group being granted *realtime* (or *rtprio* in PAM terminology) privileges. If *rtprio* is being granted to a group, then you should make sure that **you are in that group**. For instance, if the group *audio* is being granted realtime privileges then execute

```
cat /etc/group | grep audio
```
- And look to see that your userinput is listed as a member of the *audio* group. If not, execute

```
usermod -a -G audio youruserinput
```
- as root to add your user to the audio group.

If *limits.conf* does not exist... Then your system does not use [PAM](#).

If no group is given real time priority, create a group and add yourself to the group. This must be done with root privileges:

```
groupadd realtime
usermod -a -G realtime yourUserName
```

Then add `realtime` priority permissions to the group (and accordingly any user in that group):

```
@realtime
@realtime
```

```
hard
soft
```

```
rtprio
rtprio
```

```
20
10
```

JACK does not indicate realtime mode in the Status window, on a system not using the PAM security model.

- Click the *Setup* button to open *preferences*. Activate *Realtime* in the left column, and then restart QJackCtl while you configure your system further.
- To enable realtime priority for your user applications, install `set_rlimits` from your repository or directly from physics.adelaide.edu.au/~jwoithe
- As root, open `/etc/set_rlimits.conf.new` in your favourite text editor. Add two lines:

```
@audio /usr/local/bin/jackd nice=-1 rtprio=80 memlock=100000 @audio /usr/local/bin/qtractor nice=-1 rtprio=80 memlock=100000
```
- Save the file as `/etc/set_rlimits.conf`
- Create a custom launcher so that when you start your realtime applications, they are started in the syntax `set_rlimits qtractor`. In KDE, do this by right-clicking on the K Menu and edit the launch commands for QJackCtl and Qtractor. In Gnome, edit as root the `.desktop` file for the applications (ie, `/usr/share/applications/qtractor.desktop`) such that the `Exec=` line executes `set_rlimits qtractor`.

JACK does not start when Qtractor is launched.

1. Your distribution may not have configured JACK to automatically start upon Qtractor's launch.
2. The easiest workaround for this is to launch QJackCtl manually first, and then launch Qtractor.

Quick Start

Diving into Qtractor ; a quick start guide for the basic workflow.

Getting Sound Into Qtractor

Getting started with basic music production in Qtractor is simple as long as all of the requirements are installed. Review the Installation chapters to ensure you've installed all the necessary components before beginning.

As long as you've done that, you'll find Qtractor a robust digital production environment with plenty of knobs and dials to tweak.

1. Start the QJackCtl application (recommended), or start `jackd` from a shell. Once launched, press the *Start* button to start JACK.
2. Launch Qtractor from your applications menu, dock, or launcher.
3. It's good practise to **start every session by saving**. It might seem strange to save an as yet empty session, but it's better to save an empty session than to start creating your masterpiece and have data files and MIDI files scattered all throughout your hard drive. Saving first is a good way to instantiate an environment in which you can keep all of your files and sounds organized and consolidated.

To save, click the *File* menu and select *Save As*.

In the *Name* field, name your session. For the *Directory* field, click the directory icon to create a new, empty directory for your session files and click *OK*.

Save your session by clicking the *OK* button; a Save Session dialogue will open so that you can now navigate to the directory you've created and name your session file, which will appear as a `.qtr` file. Click the *Save* button to confirm.

1. Now you're ready to start producing! Obviously there are different means to creating music in Qtractor, but nearly each one would create with creating a new track. To create an empty track in your Qtractor workspace, click on the *Track* menu and select *Add Track*.

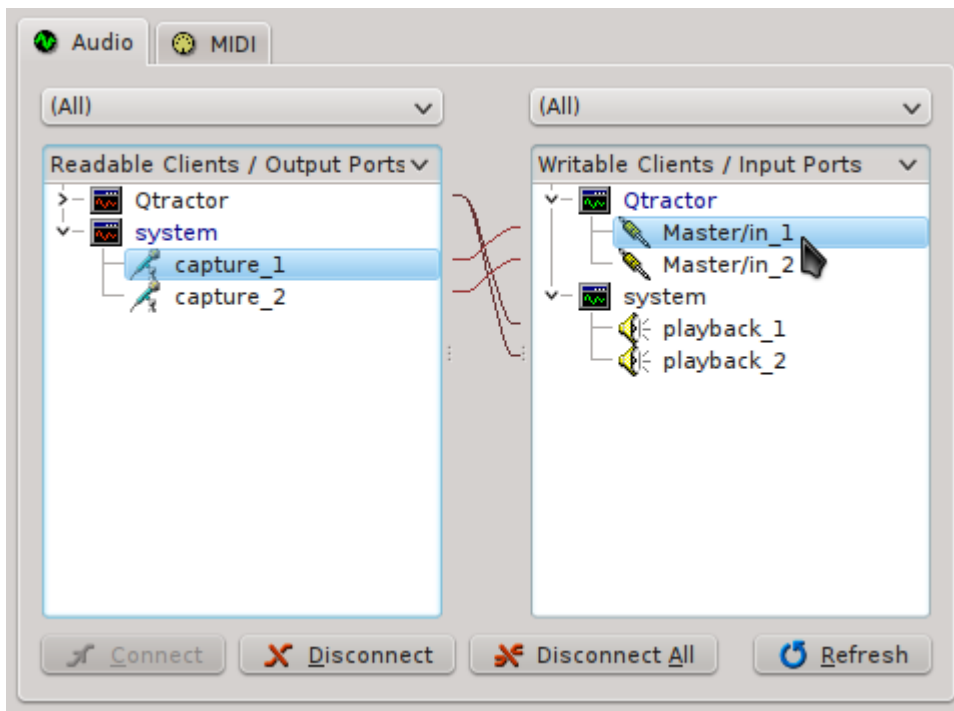
In the *Track* dialogue, name your track in the *Name* field. For *Type*, choose between *Audio* or *MIDI*, depending on what kind of data you wish to use in the track. If you have a specific set of inputs and outputs that you wish to use for the track, you can set that here, or you can leave it on the default and manipulate it later as needed.

Likewise, if you know the MIDI bank or patch selections, you can set it here, or leave it as default values and configure it later. Click *OK* to create the track.

Now that you have an empty track, you should put some sound into it. You can do this in three ways:

- Record audio directly into the track - to record directly into Qtractor, you must define your input device in your central patchbay (ie, QJackCtl). To do this, choose *View menu > Windows > Connections*.

In the left column, choose the source of sound (a device that, from Qtractor's perspective, is outputting sound, hence the "output" label). Your microphone exists as part of your computer system, so choose the *System* category, and click a capture device there.



Now choose the sound source's destination in the right column. You are routing the sound from the microphone into Qtractor, so choose the Qtractor category and select the corresponding input port, such as *Master/in_1*.

- Click the *Connect* button if they are not already connected. **Repeat this process for the right channel**, and then close the window.
- In the track label on the left of the Qtractor window, click the *R* button to arm the track for recording.
- Click the *Record* button in the top menu bar to activate recording mode.
- Click the *Play* button in the top menu bar to begin.
- Insert an existing audio file into the track - if you're using pre-recorded material from a live performance, a sound booth session, or a sample or loop collection, then there is no need to set up a recording source such as a microphone or line-in. You'll simply add files to your Qtractor session and drag them into your workspace.

To add sounds files to your session, go to the *Clip* menu and select *Import*.

Choose the file you wish to import, and it will be added into your empty track. Notice that it is also added in the *Files* panel on the right of the Qtractor window; so to add files without adding them to a track, you can right-click in the *Files* panel and choose *Add Files*.

- Play MIDI into the track - similar to recording audio directly into Qtractor, you can connect a MIDI controller to your computer and record MIDI data into a track. The MIDI data will be used to trigger a sound source ; the simplest and most direct method is to use the MIDI to trigger a software synthesizer, but of course the MIDI signal could also be routed back out of the computer into a hardware synth as well.

Assuming you are using a USB-based MIDI controller such as the Oxygen or Axiom series, or AKAI LPK series, and so on, then you must first define the source of your MIDI signal. Click the *View menu > Windows > Connections* to see your central patchbay (ie, QJackCtl).

In the left column, choose the source of your MIDI signal (a device that, from Qtractor's perspective, is outputting the signal, hence the "output" label). Your USB controller should be listed as a source, so open its listing and click the MIDI channel listed there.

Now choose the MIDI signal's destination in the right column. You are routing the sound from your hardware controller to Qtractor, so choose the Qtractor category and select the corresponding input port, such as *0:Master*.

Click the *Connect* button if they are not already connected. Close the window.

If you start playing now, you'll be sending MIDI data without hearing any feedback whatsoever, because you have not yet configured the MIDI signal to trigger sound. So that you can hear what you're playing, you must insert a plugin synthesizer into the track.

With your destination track selected, click on the *Track* menu and select *Track Properties*.

In the *Track* dialogue, click the *Plugins* tab. In the *Plugins* tab, click the *Add* button on the right to see a list of available software plugins. In the *Plugins* dialogue, select the type of plugin you wish to use with the top right button.

Software synths on Linux are usually of the DSSI variety; from the list of DSSI synths, select the one you want to play and click the *Activate* checkbox in the lower left corner.

Click the *OK* button in the bottom right corner of the window to proceed.

If you don't have any plugins installed, you can install them from your distribution or directly from their project websites.

One of the best soft synths for immediate gratification is whySynth, which bundles some nice, ready-to-use patches. You will need to restart Qtractor in order for your new soft synths to be available.

Back in the *Track* dialogue, select the *Track* tab again. In the *MIDI/Instrument* panel, choose an instrument from the top dropdown menu; using WhySynth as an example, you would select ****WhySynth_20100922 DSSI plugin****.

Choose a *Bank* and *Program*. If you're not familiar with WhySynth, you can select any of its three Banks and any of the Programs contained in them. You should be able to instantly audition the sounds on your USB controller.

When you've selected the sound you'd like to use, click the *OK* button.

- In the track label on the left of the Qtractor window, click the *R* button to arm the track for recording.
- Click the *Record* button in the top menu bar to activate Recording mode.
- Click the *Play* button in the top menu bar to begin.
- Insert existing MIDI data into the track - if you want to use pre-made MIDI files then simply add the files to your Qtractor session.

To add MIDI files to files to your session, go to the *Track* menu and select *Import Tracks > MIDI*.

Choose the file you wish to import, and it will be added into a new track. Notice that it is also added in the Files panel on the right of the Qtractor window; so to add files without immediately adding them to your workspace, right-click in the *MIDI* tab of the Files panel and choose *Add Files*.

The MIDI file you import may or may not have generic MIDI instrument assignments. Either way, you can always adjust the MIDI instrument assigned to that track via the *Track Properties* dialogue, as you would for a track into which you're recording MIDI.

Effects and Filters

You know how to create tracks and insert sound either in the form of audio files or MIDI data. In addition to putting sound into tracks, Qtractor can pipe tracks through filters and effects. The most popular format for sound filters and effect units in Linux is LADSPA, providing the free equivalent to VST, RTAS, or AU. The two ubiquitous Linux plugin packs are the Steve Harris LADSPA Plugins and the CALF Plugins, which should be available from your repository or from their respective websites.

If you have not installed them yet, install them now and then re-launch Qtractor so that it will detect them as available filters. Effects occur upon the sound itself, so they can be applied to Audio tracks as well as MIDI tracks. To place an effect on a track:

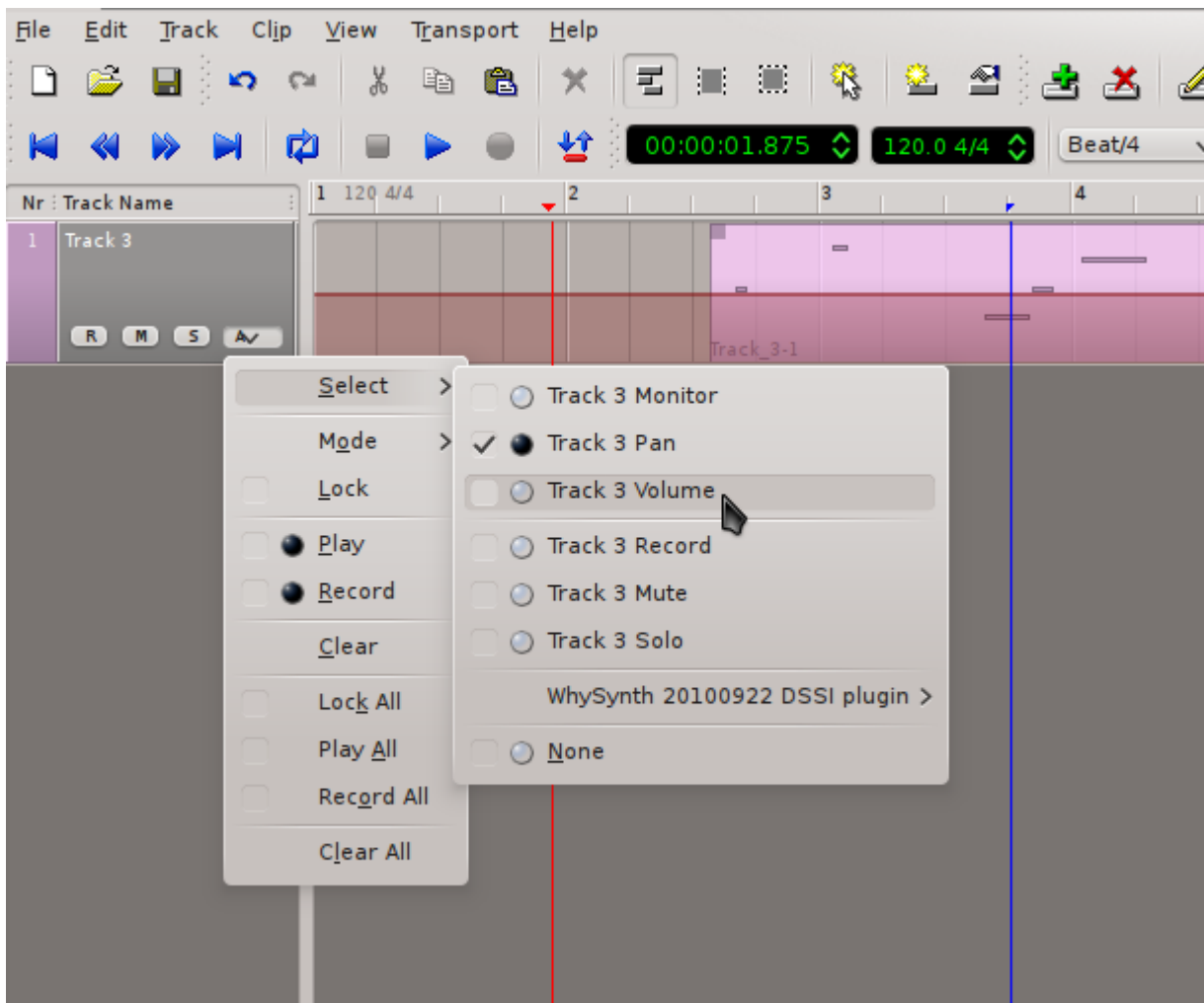
1. Select the track you want to pipe through an effect and select it by clicking the track label on the left of the Qtractor workspace.
2. Click on the *Track* menu and select *Track Properties*.
3. In the *Track* dialogue, click on the *Plugins* tab.
4. Click the *Add* button on the right of the window.
5. In the *Plugins* dialogue, choose the type of plugin you want to use with the filter button in the top right corner; more than likely, you will be using a LADSPA plugin.
6. A very common effect, of course, is reverb. Filter the choices from all the LADSPA plugins by typing “verb” into the search bar along the top. You may see one or two results, such as GVerb and Plate Reverb, so choose one of these. Place a tick in the checkbox in the lower left corner of the screen, labeled *Activate* and click *OK*.
7. Should you need to modify the reverb filter, right click on the plugin in the *Track* window and choose *Properties*. This will open the control panel for the effect, where you can change the attributes of the effect.

Automation in Qtractor

Nearly every aspect of any track can be automated in Qtractor, from the basics like volume and panning, to the very minute like the [LFO frequency](#) of a soft synth, or the levels of an effect.

To work on the automation of your tracks:

1. Click the automation button in the track control of your track.



1. Choose from the popup menu the attribute you wish to automate.
2. Notice that an overlay appears over your track, representing the normal level of that attribute. To enter automation mode, click the *Edit menu > Select Mode > Automation*. In this mode, click the automation overlay to adjust levels.
3. To differentiate different attributes that you're automating, it can be helpful to modify the colour of the automation overlay. To do this, click the automation button in the track list and choose *Mode > Colour* and pick a new shade for that automation overlay. To add automation for another attribute, click the automation button again, and navigate to *Select* to choose the next attribute. The new automation overlay appears in the default colour, and you can now modify its levels.
4. Leave automation mode by toggling off *Edit menu > Select Mode > Automation*.

Bouncing the Project

Once you've finished your song, you obviously want to export the piece so that people can listen to it without having to have your Qtractor source files. This process is sometimes called "bouncing" a track, or exporting a "mixdown" or simply "exporting" the song.

Since it's possible to have multiple sources contributing to your final product (external hardware synths feeding sound into Qtractor while soft synths play over pre-recorded audio files to the beat of a Hydrogen drum machine on another virtual desktop, for example) it would make no sense to treat the exporting process in the same way as a wave form editor (such as [Audacity](#)) would.

Qtractor's bounce process is literally to play all the sound sources in sync whilst recording what is playing back into Qtractor itself.

Different people deal with bouncing and exporting in different ways, depending on their preference, their creative needs, and the capabilities of their computers. For instance, you could bounce each MIDI track individually to an audio track and then export all audio tracks into one self-contained track.

Or you can simply set Qtractor to record and then play everything back into one audio track, which you then export.

There is no right or wrong way to do it. Either do it all in one go at the end or do it step by step.

Usually the raw power of your computer will be the deciding factor.

To perform the final export of your completed work:

- Create a new Audio track: *Track > Add Track*
- Label the track, ie, "Bounce"
- Open the *Connections* window: *View > Windows > Connections*
- In the *Connections* window, connect the *Master/Out* of Qtractor to the *Master/In* of Qtractor, such that the output of all sound managed by Qtractor is being directed to the input of Qtractor for recording.

Be sure to disable any capture device such as microphones or unused line-ins. You'll be recording all the live sound going into Qtractor, so you don't want accidental line noise, hums, or room tone.

- Set the in and out points for your recording by clicking once on the top timeline at the point you want the recording to stop; a blue transport line will be anchored where you click. Scroll back to the beginning of your project and click again in the top timeline to mark the in point with an opening blue transport bar.

Click the *Punch In/Out* button in the top menu bar to limit playback between your markers.

If you do not set in and out points, you will need to stop the recording manually.

- *Arm* your Bounce track for recording. Click the *Record* button in the top menu bar.
- Make sure that your transport (playhead) is at the beginning of your track! When ready, press the *Play* button in the top menu bar.

Bouncing a track is a realtime process ; do not use your computer while you're bouncing a track!

When you've bounced your song to its own track, save the project! /Solo. the bounce track and listen to the recording for quality assurance, and then you are ready to export that track as a self-contained, distributable file.

To export a single track to disk:

1. Select the track or tracks you wish to export. If you bounced everything into one track, then it will just be that track, which you can solo using the *Solo* button on the track controls. If

you are exporting a project consisting of only in-project audio, then leave them all activated and continue.

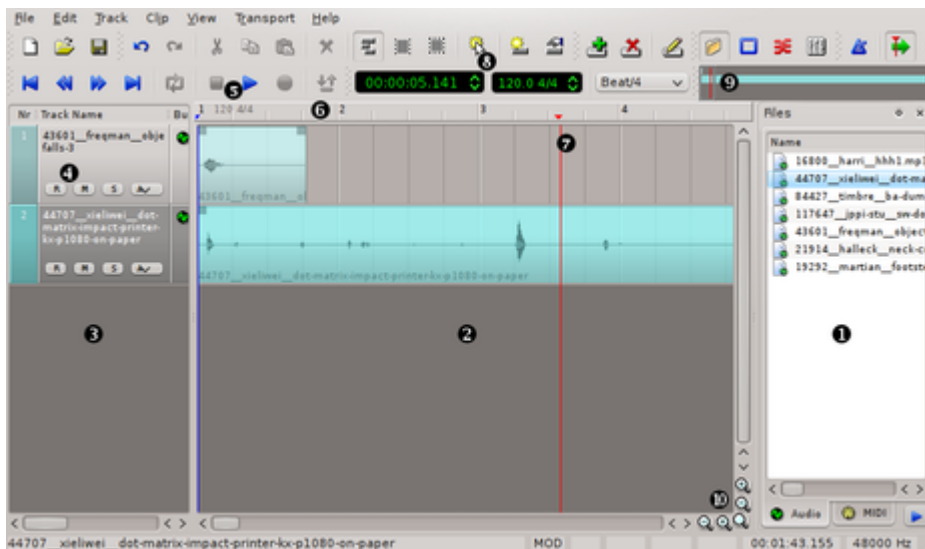
2. Choose *Track > Export Tracks > Audio*
3. In the *Export Audio* dialogue, choose a location and filename for your file. Choose the range of what will be exported; you can export the entire session, or manually set a Punch In/Out range, or specify the range in timecode, frames, or bars/beats/ticks. If you have multiple output sources, choose the appropriate source (probably your Master Output).
4. Click the OK button to begin the export.

Advanced Qtractor

How to navigate and use the many features of Qtractor for audio production.

The Qtractor Interface

Qtractor's interface will be familiar to anyone familiar with Digital Audio Workstation software.



1. *Files* - a list of audio and midi files you've imported into your Qtractor project
2. *Workspace* - a view of the audio or MIDI data contained in each track in your project
3. *Tracks* - a list of tracks and some track-specific controls and information
4. *Track controls* - titles, bus information, recording and playback toggles, and automation controls for each track in the workspace
5. *Transport controls* - rewind, forward, play, pause, record
6. *Timeline* - your music workspace's timeline in bars, measures, and timecode
7. *Transport* - the playhead
8. *Timecode* and *BPM* - counter for SMPTE timecode, and the tempo or beats-per-minute setting

9. *Overview* - the entire project in one thumbnail view
10. *Zoom* - zoom in and zoom out (horizontally and vertically) buttons

Starting a new session

The first step in so many pro applications is one that is most overlooked by new users: create and save a new session. Saving a session before actually doing any work in it seems counter-intuitive, but it's important to save first so that any and all files created or required for your work can be contained in a pre-determined location.

To create a new session in Qtractor:

1. Launch Qtractor for a fresh new session, otherwise go to the *File* menu and select *New* or use **Ctrl+N** to start a fresh session.
2. Go to the *File* menu again and select *Save As* (or *Save* if you've already been working in the opened session), or just hit **Ctrl+S**
3. In the *Session* dialogue box, name the session and provide a description as needed. Click the directory button on the right to enter a file chooser dialogue.
4. In the *Session Directory* dialogue, create a new folder in the directory of your choice; for example, you might make a **myGreatSong** folder for your session in your **~/music** directory.
5. Click the *OK* button once you've created a directory for your session files.
6. Click the *OK* button in the *Session* dialogue box to proceed
7. A *Save Session* window will appear, which is a prompt for you to save the session file itself into the destination you've just created for it. Navigate to your new folder (in this example, **myGreatSong** in the **~/music** directory)
8. Name your session file; by default it will already be named the same as the name of your session itself, but if you are versioning the file or have different notation you'd like to use, you can give it a customized name as well. The extension for Qtractor session files is **.qtr**
9. Click the *Save* button, and now your Qtractor session is saved in a self-contained, dedicated folder. This helps your project maintain its integrity and increases its portability.

Managing Audio Tracks

There are four typical scenarios for dealing with sound in Qtractor and, indeed, for most digital audio workstations:

- Import existing audio files from your harddrive, a recording device, or a loop collection
- Record audio into Qtractor via your computer's built-in microphone or an external microphone
- Import MIDI data
- Create MIDI data

All three of these methods of acquiring sound for your work can be, and very often are, combined to produce a richer audio production.

Importing Audio Files

The easiest way to get sound into Qtractor is to import an audio file from your harddrive. Whether you are using sampled loops to construct a new musical piece, or importing a performance transferred from a recording device, importing audio in this way does not involve recording sound directly into Qtractor.

To bring audio into your project file, right-click in the *Files* pane and select *Add Files* or use **Ctrl+F**. Choose what file or files you wish to import from the file chooser window that appears. To place an audio file in a track, drag and drop it from the Files panel into the workspace. You can add it to an existing track, or drop it directly into empty workspace and a new track will be created automatically.

Recording Audio

Recording audio into Qtractor requires a microphone and at least one audio input channel. Many laptops and webcams have built-in microphones, so in theory you could use this as an input source, but for best quality, purchase an external microphone and use it as your audio capture device.

For simultaneous multitrack recording, rarely will the soundcard that was bundled with your computer be sufficient. Almost all soundcards embedded on motherboards are set to mixdown the input signals to a stereo mix. If you wish to record three separate musicians at the same time, each to a separate track in Qtractor, you will require a soundcard with separate dedicated inputs.

The way that Linux displays available sound inputs and outputs can be daunting at first, until you understand the logic behind it. A Linux system displays sounds devices the same way it displays hard drives and available network interfaces: the first device (regardless of actual inputs or outputs available via that device) is labeled **hw0**, the next **hw1**, the next **hw2**, and so on.

It is safe to assume that **hw0** would be the built-in sound card on the system; being embedded in the motherboard would certainly qualify it as the first available sound device.

So, **hw0** represents, in almost every case, your built-in sound card. **hw1** might represent, for instance, a webcam that you keep plugged into your desktop, and **hw2** could represent, perhaps, a USB microphone or a USB interface that you've plugged in.

You can usually determine which device is which by looking at the vendor name associated with the Hw labels; if I have two devices plugged into my system and one is labeled **Blue** and the other **H4**, then I would know from these terms that one is my Blue USB microphone and the other my Zoom Studio H4n.

With regards to outputs, these can sometimes be confusing due to the many possible ways you may wish to output your sound. Typically the stereo mix of you system sound is available as the first two output devices.

If you have more than just two speakers and you wish to split your sound to each, then utilize the outputs labeled appropriately (Front, Center, and so on). If the sound input and output labels confuse you, take a few minutes to learn them by playing sound on your computer and plugging speakers into each output on your computer.

For the inputs, plug a microphone into your different inputs and see where they are received and how they are labeled. It won't take long before you understand the logic behind the labels, and you'll be able to use Qtractor all the more fluidly.

Analogue vs USB Inputs

There are many kinds of microphones, each intended for certain kinds of sounds and situations, but on a purely technical level, without the question of aesthetics and microphone design, there are only about three scenarios you will encounter:

- Microphones with an 8th- inch (also called a mini) jack
- Microphones with a quarter-inch or XLR jack
- Microphones with USB connectors

The recording process is different, depending on your input type.

Recording from Line-In

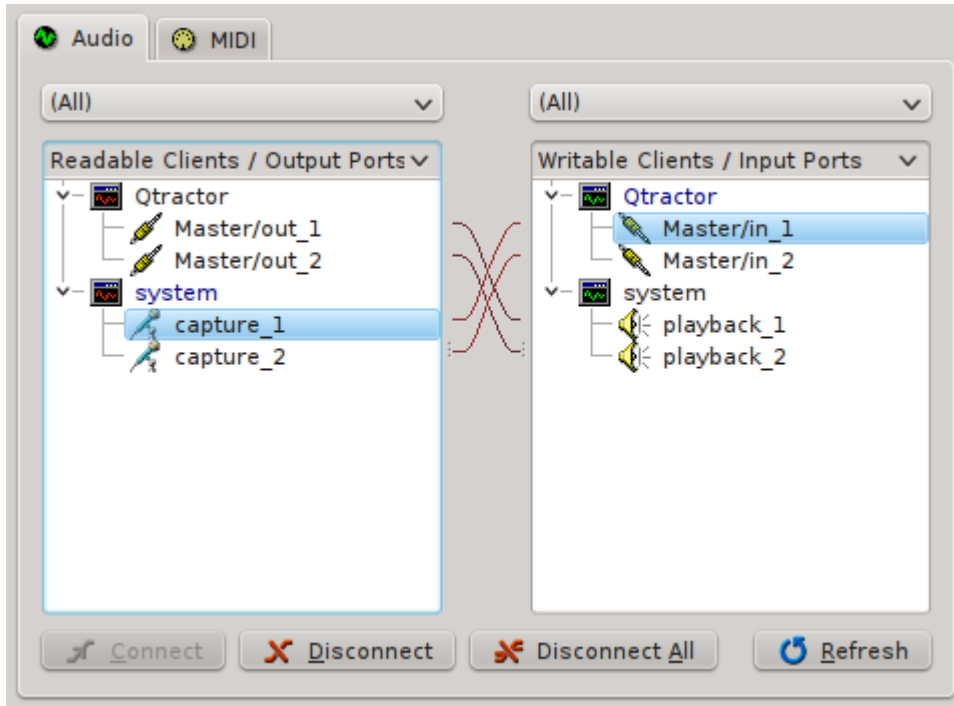
If your microphone has an 8th-inch jack, then you can plug it directly into the line-in of your computer. No external sound interface is required. Your built-in sound card is JACK's default input, so no changes are necessary.

If your microphone can easily and cleanly adapted to the standard 8th-inch input with a cable or a simple plug adapter, then you can use this method of recording, as well.

To record from the line-in of your computer into Qtractor:

- Go to the View menu and select Windows > Connections to verify that the Capture devices on your System are routed to the Master/In of Qtractor, and the Master/Out channels are routed to the Playback channels of your System.

Recording from USB Audio



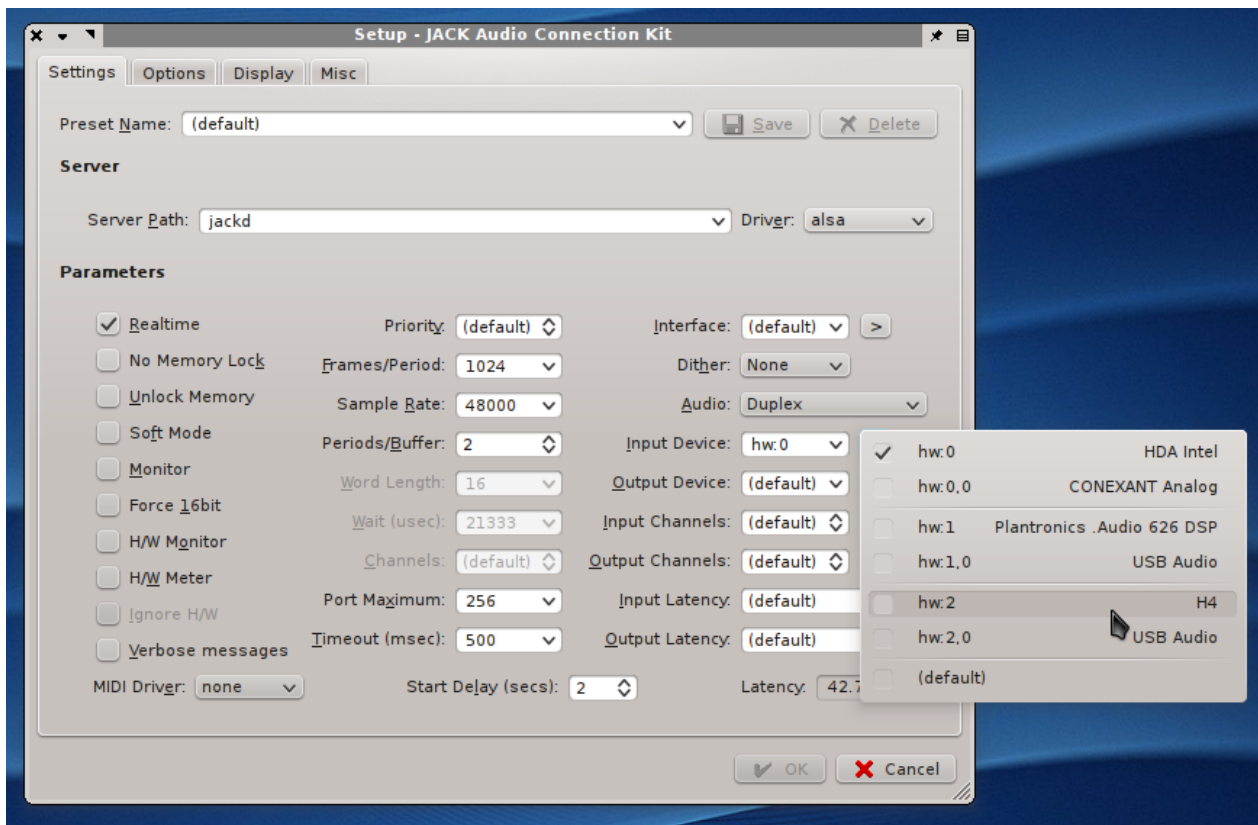
- Go to the Track menu and choose Add Track or use Ctrl+Shift+N. In the Track dialogue box, give the track a name, set the Type to Audio, and set the Input/Output to Master. Click the OK button to proceed.
- Arm the new track for recording by clicking the R button in the track listing on the left of the Qtractor window. This sets the destination for the recorded sound.
- Click the Record button in the top toolbar
- Click the Play button in the top toolbar

If your microphone has a quarter-inch or XLR jack and you choose not to use a plug adapter, then you will need an external sound interface. External interfaces are available from M-Audio, Fostex, Zoom Studio, and others; they serve as an intermediate converter from your input device and the USB port of your computer.

Similarly, a USB microphone plugs directly into the USB port of your computer.

If you input sound through USB, then (obviously) you are utilizing a different interface than your computer's built-in sound card. This must be set via QJackCtl for appropriate sound routing to occur:

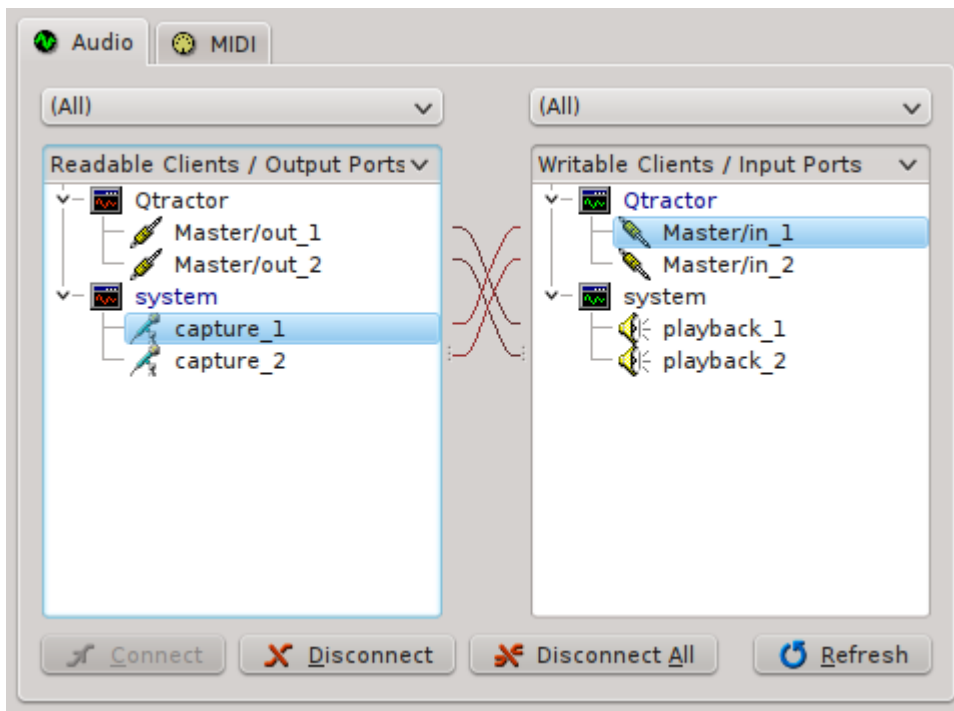
- In QJackCtl, stop the sound server by clicking the Stop button
- If you have not already plugged in your USB audio interface or USB microphone, then do so. Make sure it's on.
- Click the Setup button. In the Settings tab, locate the Input Device setting and click the > button to see your choices.



- Click the Start button to activate.

Now your USB interface, whether it is a single USB microphone or a 4-channel Audio-to-USB conversion box, is managing your input sources. Plug your XLR microphone into your USB interface, set the interface's input source as your XLR jack, and then create a new track in Qtractor and begin the recording:

- Go to the View menu and select Windows > Connections to verify that the Capture devices on your System are routed to the Master/In of Qtractor, and the Master/Out channels are routed to the Playback channels of your System.



- Go to the Track menu and choose Add Track or use Ctrl+Shift+N. In the Track dialogue box, give the track a name, set the Type to Audio, and set the Input/Output to Master. Click the OK button to proceed.
- Arm the new track for recording by clicking the R button in the track listing on the left of the Qtractor window. This sets the destination for the recorded sound.
- Click the Record button in the top toolbar
- Click the Play button in the top toolbar

Managing MIDI Tracks

MIDI is a versatile format for triggering sounds; it can be used to trigger hardware synths external of your computer, software synths that are applications on your computer, samples, loops, automation functions, and more. There are basically three ways to get MIDI data into your Qtractor session:

- Import MIDI data from an existing file
- Enter MIDI data into a matrix editor (sometimes called a “piano roll” or “grid view”)
- Play in MIDI data via a USB MIDI keyboard controller

Before beginning with MIDI, you should make sure that you have some software synthesizers installed on your system. A popular format for soft synths on Linux is DSSI, which is akin to VST or AU on other platforms. LV2 is also quite popular, although in practice the bulk of LV2 plugins you’ll find are audio effects units (reverbs, tape delays, EQ, and so on).

One of the easiest soft synths is WhySynth; it features enough presets to provide instant gratification, and enough control over the raw sound to become a permanent fixture in your studio. You can install WhySynth from your distribution’s repository or directly from If you’ve just installed soft synths now, then you should relaunch Qtractor so that it will detect the newly available plugins.

Importing MIDI Data

Since MIDI data consists of nothing more than digital signals to activate and deactivate sounds at certain times, MIDI files are small and easily distributed online. A simple internet search for MIDI files will result in thousands of MIDI files for nearly any song you can name. Taken alone, the MIDI file is useless. Imported into Qtractor and assigned instruments, the files come to life like a player piano. The MIDI file you import into Qtractor could be from the internet, or it could be a MIDI file you yourself created on any given MIDI input application, or even from data dump from a MIDI-capable hardware synth. In this example, we will use one from the internet:

1. Download a MIDI file, such as [JS Bach "Invention in c minor, BWV 773 \(Canon\)"](#) and save it to your hard drive. Preferably, in your Qtractor's session directory, to keep your project self-contained.
2. In Qtractor, go to the Track menu and select Import Tracks > MIDI. Alternately, you may click on the MIDI tab of the Files panel on the right of the Qtractor window and right-click or use Ctrl+F to import the MIDI file to your files panel and not automatically add its tracks to your session. If you do this, then you'll need to manually drag the MIDI file into the workspace in order for the data to appear as tracks.
3. The MIDI tracks will appear in your Qtractor session, in this example's case it is even preprogrammed for General MIDI instrumentation. Imported tracks will often have embedded General MIDI instrument assignments. General MIDI was an effort to standardize a set of 128 instruments to provide similar results in playback across all systems.

If you have a soft synth bank compliant with General MIDI, then you can press the Play button in the top toolbar of Qtractor and you will get a good approximation of how the original author of the file wanted the tracks heard.

You can also use the list of Generic MIDI instruments found in the appendices of this book to guide you in the instrumentation of an imported file, making the sound of the tracks truly your own.

To assign instruments to a track:

1. right-click on the track title and select Track Properties from the contextual menu.
2. In the Track dialogue box, click on the Plugins tab.
3. In the Plugins tab, click the Add button on the right to see a list of available software plugins.
4. In the Plugins dialogue, select the type of plugin you wish to use with the top right button. Software synths on Linux are usually of the DSSI variety; from the list of DSSI synths, select the one you want to play and click the Activate checkbox in the lower left corner.
5. Click the OK button in the bottom right corner of the window to proceed.
6. Back in the Track dialogue, select the Track tab again. In the MIDI/Instrument panel, choose an instrument from the top dropdown menu; using WhySynth as an example, you would select WhySynth_20100922 DSSI plugin.

7. Choose a Bank and Program. If you're not familiar with WhySynth, you can select any of its three Banks and any of the Programs contained in them. You should be able to instantly audition the sounds on a USB controller.
8. When you've selected the sound you'd like to use, click the OK button. Repeat these steps for each track to wish you want to assign an instrument, incrementing the MIDI channel you use for each new instrument you assign. Press the play button in the top menu bar to hear the MIDI file play.

Note that you must increment the MIDI channel used, or keypresses on your MIDI controller will be using the same MIDI channel to trigger different sounds. In otherwords, if Track 1 and Track 2 are both using MIDI Channel 1, then a note played on Track 2 will also trigger the instrument assigned to Track 1.

Creating MIDI data with a MIDI controller

A MIDI controller is a piece of hardware to help you play the software-based synthesizers in your computer. Typically, a MIDI controller is a piano keyboard with a few octaves and no built-in sounds of its own. It usually plugs into the USB port of your computer and, once configured, will pass on any key press to Qtractor, allowing you to play a software synth in realtime as well as record the keypresses themselves.

MIDI and USB are two technologies that, fortunately, have been kept universal enough that you almost don't need to question whether any given USB MIDI controller will work with your system. There are perfectly capable USB MIDI controllers being offered from vendors like Akai, Roland/Edirol, gidesign/M-Audio, and many others. If you require realistic, weighted keys with high resolution touch-sensitivity then look at the upper price-range of controllers and go to a music store to audition them.

If all you require are a few octaves of piano keys so that you can get a tune into the computer quickly and easily, then the basic controllers will be enough.

To use your USB controller to input MIDI data:

1. Plug in the USB MIDI controller to your computer
2. Launch QJackCtl and start it
3. Launch Qtractor and click on the View menu, and select *Connections* from the *Windows* category
4. In the Connections window, click on the MIDI tab. Listed on the *Readable Clients / Output Ports* column, find your USB controller device. Open its category to see the available outputs.

If you do not see your USB controller listed in the left column, check to ensure that your controller is powered on (some require external power, others are powered by the USB port), and check the cable connection. If you still cannot see the device in the left column, make sure your computer sees the controller by checking `dmesg | tail`. If your computer registers the device but you are not seeing it in Connections, try restarting both QJackCtl and Qtractor.

1. Connect the output of your MIDI device to Qtractor in the Writable Clients/Input Ports column. Qtractor is now configured to receive MIDI events from your USB controller. To play your composition (or parts of your composition) into Qtractor, you will need an empty

track. To begin with, create an empty track by selecting *Add Track* from the *Track* menu, or by clicking the *New Track* button in the top toolbar.

To assign instruments to a track

- Right-click on the track title and select *Track Properties* from the contextual menu.
- In the *Track* dialogue box, click on the *Plugins* tab.
- In the *Plugins* tab, click the *Add* button on the right to see a list of available software plugins.
- In the *Plugins* dialogue, select the type of plugin you wish to use with the top right button.

Software synths on Linux are usually of the DSSI variety; from the list of DSSI synths, select the one you want to play and click the *Activate* checkbox in the lower left corner.

- Click the *OK* button in the bottom right corner of the window to proceed.
- Back in the *Track* dialogue, select the *Track* tab again. In the *MIDI/Instrument* panel, choose an instrument from the top dropdown menu; using *WhySynth* as an example, you would select *WhySynth_20100922 DSSI* plugin.
- Choose a *Bank* and *Program*. If you're not familiar with *WhySynth*, you can select any of its three Banks and any of the *Programs* contained in them. You should be able to instantly audition the sounds on a USB controller.
- When you've selected the sound you'd like to use, click the *OK* button. Now that you've create an empty track and assigned it a soft synth, you can record MIDI data:
- Create an empty clip in your destination track by choosing *New* from the *Clip* menu, or by clicking the *New Clip* button in the top toolbar
- In the *Session* window, type in a name for your MIDI clip, choose the Directory into which you want to save the clip, and provide a description for yourself as needed. Click the *OK* button when finished.

Unlike pre-recorded loops or sounds, which already exist on your harddrive, a new MIDI clip doesn't yet have a home. Some music programs bundle your MIDI data into your project file, which can make it difficult to share one MIDI loop between projects. But *Qtractor* saves all MIDI tracks as independent files, meaning you can easily re-use them.

- Arm your track for recording by clicking the *R* button in the track control
- Click the *Record* button in the top menu bar to prepare the track for recording. Click the *Play* button to start the transport recording
- Every key press you make on your USB controller will be recorded in realtime. Depending on the feature set of your controller, velocity, pitch, and other data may also be recorded

Creating MIDI data with the matrix editor

If you have no USB MIDI controller, you can still create MIDI data using *Qtractor's MIDI Editor* (sometimes called a "matrix editor" or "piano roll" in other music programs). This enables you to

mark on a grid when a note should be triggered, how long it should be sustained, and at what velocity it should be played.

Broadly speaking, the steps are:

1. Create a destination *Track*
2. Create a MIDI *Clip*
3. Enter the data in the *MIDI editor*

To create a new track select *Add Track* from the *Track* menu, or click the *New Track* button in the top toolbar.

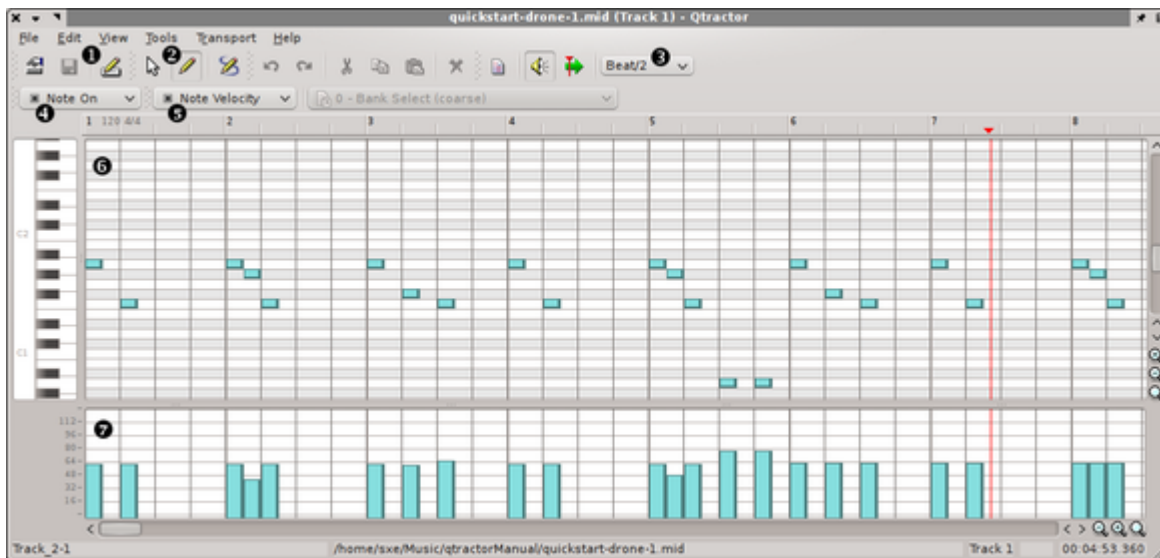
To assign instruments to a track

- Right-click on the track title and select *Track Properties* from the contextual menu.
- In the *Track* dialogue box, click on the *Plugins* tab.
- In the *Plugins* tab, click the *Add* button on the right to see a list of available software plugins.
- In the *Plugins* dialogue, select the type of plugin you wish to use with the top right button. Software synths on Linux are usually of the DSSI variety; from the list of DSSI synths, select the one you want to play and click the *Activate* checkbox in the lower left corner.
- Click the *OK* button in the bottom right corner of the window to proceed.
- Back in the *Track* dialogue, select the *Track* tab again. In the MIDI/Instrument panel, choose an instrument from the top dropdown menu; using WhySynth as an example, you would select WhySynth_20100922 DSSI plugin.
- Choose a Bank and Program. If you're not familiar with WhySynth, you can select any of its three Banks and any of the Programs contained in them. You should be able to instantly audition the sounds on a USB controller.
- When you've selected the sound you'd like to use, click the *OK* button. Now that you've create an empty track and assigned it a soft synth, you can input MIDI data:
- Create an empty clip in your destination track by choosing *New* from the *Clip* menu, or by clicking the *New Clip* button in the top toolbar
- In the *Session* window, type in a name for your MIDI clip, choose the *Directory* into which you want to save the clip, and provide a description for yourself as needed. Click the *OK* button when finished.

Unlike pre-recorded loops or sounds, which already exist on your harddrive, a new MIDI clip doesn't yet have a home. Some music programs bundle your MIDI data into your project file, which can make it difficult to share one MIDI loop between projects. But Qtractor saves all MIDI tracks as independent files, meaning you can easily re-use them.

- Click on the clip to select it, and choose *Edit* from the *Clip* menu to open the clip in the *MIDI editor*. The MIDI Editor is very intuitive, with simple but effective tools in the toolbar, such as an *arrow* to select and move notes, a *pen* tool to draw note into the grid,

and a *disk* icon to save your work. Vertically, the grid corresponds with **notes** on the chromatic keyboard. Horizontally, the grid corresponds with **beats** per measure.



Properties of the MIDI Editor workspace, Save, and Track Properties

- *Arrow* - to move notes, increase and decrease note length, and selecting *Pen* (Edit On) - to draw note into the grid Edit Draw - dynamically draw multiple notes with a click-and-drag of the mouse
- Snap/Beat menu - determines the resolution of drawing and editing notes. In a 4/4 project, for instance, a setting of Beat/2 would cause a pen tool click to produce an eighth note, while *Beat* would produce a quarter note, and so on. Regardless of the default note length, a note block can be extended by clicking and dragging the border of the note to increase (or decrease) its length. The resolution of changes to the length of a note is also determined by the Snap/Beat menu.
- *Notes Type* setting indicates that the notes you draw in the grid represent the length of the note. The Key Press setting indicates that the notes you draw in the grid represents a MIDI event without a defined duration, such as triggering an external MIDI sequence to begin playing.
- The *Value* menu lets you control note velocity, program changes, pitch bending, and other extra MIDI data.
- The main editor grid represents the notes being played. Use the *pen* tool to draw notes into the grid according to the keyboard on the left of the window. Whether you are drawing in notes or only key presses can be controlled by the *Notes Type* menu.
- The value grid controls extra MIDI data, such as velocity, program changes, pitch bends, and so on. The data represented here will change according to the Value menu setting. The MIDI Editor can be invoked either for raw entry or to modify existing MIDI data that you've either

imported or played into Qtractor.

Additional MIDI Editor Features

In addition to input and editing tools, the MIDI Editor features functions to make it easy to modify your composition, such as **Quantization**, **Transposition**, **Timeshifting**, and more.

Customizing the MIDI Editor View

To access these features, use the *Tools* menu in the MIDI Editor window. If no notes are selected, then the Tools will be unavailable; select a note or block of notes with the *arrow* tool to make them available.

- *Quantize* - allows you to automatically structure notes in stricter uniformity with your time signature. This is especially helpful if you've played the notes in and were not perfectly on the beat. You can quantize notes such that the notes occur on the beat (or one 16ths of the beat, or 8ths, or so on), such that the duration of notes are extended to be on the beat (or divisions thereof), such that notes play with some degree of "swing", and even such that the notes played match a specific scale from Minor to Major to the extremely obscure.
- *Transpose* - moves the block of selected notes up or down some number of steps.
- *Normalize* - adjusts note velocity by either a percentage or an absolute value. A percentage value will adjust the velocity of the notes equally by the percent given in relation to the original velocity, while an absolute value adjusts velocity of each note to that value regardless of original velocity.
- *Randomize* - provides random changes to the Note, Time, Duration, or velocity (Value). Adjust how extreme the changes will be with percent values.
- *Resize* - allows you to control the duration of notes to any number of beats (or divisions thereof), or the Velocity to any value.
- *Rescale* - changes the selected notes by some percentage; you may alter the time that the notes are triggered, the duration for which they sound, and the velocity at which they are played.
- *Timeshift* - alters the timing of the selected notes on a curve, such that the acceleration from the beginning note to the ending note is either increased or decreased.

In order for Qtractor to know how to accelerate the timing of the notes, you must define a range of time for the acceleration to occur. Do this with the *head* and *tail* markers (the blue markers in the timeline above the MIDI Editor grid). Mark the out point (tail) of your range by clicking on the timeline, and then set the in point (head) of the time range by clicking somewhere to the left of the original marker. Once the range is defined, select the notes you wish to **timeshift** with the arrow tool. Open the *Tools* menu and select *Timeshift*. Use the slider to either cause an acceleration in your selected notes over the course of the defined time range, or a deceleration, and then click *OK* to commit the change.

The *View* menu provides customization for how the MIDI Editor is laid and how the notes are presented to you. The default layout is clean and pleasingly minimalist, but take a look at some of the other options to see what works best for you:

- *Menubar* - turns off the top menu bar; use **Control+m** to toggle it on or off.
- *Statusbar* - turns off the status bar at the bottom of the window.

- *Toolbars* - defines what icon toolbar is visible at the top of the window; choose from *File* (file opening and saving icons), *Edit* (cut, paste, undo), *View* (preview modes, snapping quantization), *Transport* (fast forward, rewind), and *Scale* (key signature and scale information).
- *Windows* - toggles on or off event information, a panel which will provide detailed information on each MIDI event in the MIDI Editor grid
- *Tool Tips* - toggles whether tool tips are visible
- *Note Duration* - defines whether the note's duration is reflected in the length of the velocity bars at the bottom of the grid
- *Note Color* - assigns differing colours to each note; if this is not active, all notes appear as one colour.
- *Value Color* - assigns the colours of the corresponding notes to the velocity bars; helpful when you have many overlapping notes but want to adjust their velocities.
- *Zoom* - zoom in or out on the grid.
- *Snap* - define what division of the beat your notes snap to, or turn snapping off entirely.
- *Scale* - change the key signature or type of scale being used.
- *Refresh* - for the MIDI Editor to redraw in the event of latent images.
- *Preview Notes* - turn on or off whether you hear the notes of the scale as you input or move notes along the grid.
- *Follow Playhead* - define whether the MIDI Editor grid scrolls with the playhead when you are playing the track.

Qtractor Options and Preferences

Qtractor's preferences can be accessed via the View menu, via the Options selection. The options are divided into tabs:

- *General* - contains options for overall preferences, such as file formats of project files, behaviour of windows and confirmation requests, and the transport
- *Audio* - determines how Qtractor stores newly recorded audio (ie, microphone or line inputs, as well as re-routed sound within its own mixer), and whether an audible metronome is used
- *MIDI* - controls how MIDI data is stored, captured, and what defaults it uses for quantization, playback, and metronome
- *Display* - controls whether your timeline uses timecode, frames, or beats, some default colours and fonts, and where log files are stored
- *Plugins* - the paths to folders containing your plugins. An empty list is OK, since Qtractor defaults to the typical locations such as *usr/lib* and so on, but if you have customized your system heavily then you may need to specify the locations of your synths, effects, and plugins here.

Soft Synths

Strictly speaking, soft synths are not a part of Qtractor. However, it is common to use soft synths within Qtractor. Acquiring free software synthesizers online is a simple matter of locating them online at sourceforge.net, ccrma.stanford.edu, your distribution's repository, or the project's homepages directly.

Software synths for GNU Linux come in a few varieties:

- Independent synth applications that launch separately from Qtractor but that plug into JACK so that the synths can be used as sound sources within Qtractor. One such example is Qsynth, written by the same programmer as QJackCtl and Qtractor.
- DSSI Plugins which are launched and controlled entirely from within Qtractor
- LV2 Plugins which are launched and controlled entirely from within Qtractor
- VST Plugins; the format from Steinberg does in fact support Linux technically but the number of Linux-native VST synths are few

Installing Soft Synths

Installing any soft synth, whether it is a plugin or stand-alone application, is done through your distribution's package manager (yum, apt, slackbuilds, pacman, emerge, ports, and so on) or from the installers on the project's website. Once the synth is installed, you may launch it and try it out; if it's a stand-alone application, then it can be launched like any other application on your system.

If it's a plugin for Qtractor, then launch Qtractor and create a new *MIDI track*, choosing your new soft synth as the MIDI Instrument as described in Chapter Managing MIDI Tracks To verify that Qtractor knows where all of your soft synth plugins are located, open the *View* menu and select *Options* and choose the *Plugins* tab in the *Options* window.

In the *Plugins* tab, define any non-standard paths to DSSI, LADSPA, LV2, or VST plugins on your system by selecting the type of plugin from the dropdown menu on the left and then adding the path with the *Add* button on the right. When finished, click the *OK* button to save your changes.

Using Soft Synths as Plugins

Using soft synths as DSSI or LV2 plugins is probably the easiest model, as it does not require any additional routing of sounds through JACK. The synth is integrated into Qtractor's interface, the sound is directed to the appropriate track, and the fact that you are using a separate application is almost entirely abstracted away from you.

From the Chapter Managing MIDI Tracks, you already know how to create a *MIDI track* and assign a synthesizer plugin to that track. Most soft synths, however, have some level of control over the sounds they produce, which you can access by editing the properties of the synthesizer:

1. Open the *View menu > Windows* and select *Mixer*.
2. The Mixer window shows all available tracks; on the far left are the *Master Inputs* and on the far right are the *Master Outputs*; in the middle are the tracks in your project's

workspace. In the top plugin list of your track, right click on the soft synth currently associated with the track and choose *Properties* from the contextual menu.

3. In the window that appears, notice the three buttons in the top right corner:
4. *Params* toggles the view of available attributes you can modify (such as oscillators, resonance, envelopes, and so on)
5. *Edit* toggles whether the plugin is editable
6. An active plugin is “on” and will play sounds when receiving MIDI signals. If a plugin is not active, it will not produce (or affect, if it’s an effect unit plugin) sound if you do not see editable parameters in the synthesizer window, click the *Params* button.
7. Change the parameters you wish to change using the controls the soft synth provides. If you create a patch that you want to save, type a name into the *Preset Name* field in the upper left corner and click the *Save* button and your settings will be added as a Preset in the *Preset* dropdown menu.

Whether you use DSSI, LV2, or VST plugins, the process for using a soft synth or effect is the same. All parameters provided by the soft synth’s interface can also be automated such that attributes like LFO or resonance can be programmed to change over time during playback. For more information on automation, see the Chapter Automation.

Using Stand-Alone Synths

The other kind of software synth you might use in your Qtractor workflow are stand-alone synthesizers such as Qsynth and amSynth. Installing these stand-alone synths from your distribution’s repository or from the projects’ websites. Launch QJackCtl first, and then launch the software synths the same as you would any other application on your system.

Keep in mind that if your distribution requires you to set your own realtime permissions, you will want to add the stand-alone software synths to your RT list.

Once you’ve launched your software synth, it will become a source for sound and a destination for MIDI signals in the Qtractor. Verify that your settings are correct:

1. Select the *View* menu in *Windows > Connections*
2. In the *Audio* tab appear all possible sources of sound. If you have launched QSynth or amSynth, for example, they would be listed in the left column. Also listed will be Qtractor itself, since it also produces sound, and your computer’s sound input (labeled as *System > capture*) In the right column are the sound destinations available. At the very least, you should see Qtractor’s *Master In* and your computer speakers (labeled as *System > playback*) By default, the sound from Qtractor *Master Out* is patched to *System playback* so that you can hear, from your speakers, everything that Qtractor’s mixer manages.
3. This is where the logic of audio engineering as well as the workings of your own studio setup should be considered. The sound produced from your stand-alone synth can logically be patched into either:
4. Qtractor *Master In* so that the sound produced by the synthesizer can be recorded back into an empty audio track in Qtractor

If you're not sure what kind of setup you have, then choose this method.

- *System playback* if you have an external mixer or recording device that will be recording the audio being produced by the MIDI-driven synths.
- To patch the sound from the synth to its destination, click the left channel of the synth output, and then the left destination channel, and click the *Connect* button. Repeat this for the right channel if stereo is required.

This takes care of routing the sound; now you must route the MIDI signals controlling the synthesizer. As with the routing of sound, there is no “right” way to route the MIDI signal; you must be somewhat familiar with the equipment and workflow you are implementing and decide for yourself how you want it to all work together.

- In the MIDI tab appear all possible sources of MIDI signals. At the very least, your stand-alone soft synth will be listed in the left column. If you have a USB MIDI controller, it will also be listed. Qtractor itself and a MIDI Through channel are also listed. In the right column are the sound destinations available. At the very least, you should see Qtractor's Master and the MIDI Through channels, and the soft synth you are running.

As with audio routing, there are many ways to design your production process:

- The MIDI signal from Qtractor should be patched into the soft synth(s) that you are running so that the data in your MIDI tracks will trigger the sounds of those synths. If you are using an external USB MIDI controller, you might want to route the signals generated from it into the Qtractor destination so that you can play MIDI into Qtractor MIDI tracks (which in turn gets passed onto the soft synth, which plays sound back into Qtractor and/or into your computer speakers).

If you are not sure about what you want, this is probably the method to use.

A Bit about MIDI

- Alternately, you could route the MIDI signals from your controller straight into the soft synth. This would trigger the synth directly, playing sounds which could be routed into Qtractor for live recording.

This method does not record any MIDI data.

To patch the MIDI signal from the source to its destination

- Click the MIDI source in the left column
- Then the destination in the right column
- Click the Connect button
- Repeat this for all MIDI signals that need routing

If you're not sure what signals need routing, then you probably need to route Qtractor to your synth. If you are using a USB MIDI controller then you also need to route your controller to Qtractor.

All signals have been patched now, so playing MIDI into Qtractor should cause your external synthesizer to sound.

If you have had limited experience with MIDI, it's important to have a basic understanding of how MIDI talks to the devices it controls.

MIDI was originally developed for hardware, with the only software involved being MIDI sequencers, which were as frequently embedded systems as they were traditional software applications like the modern DAW's we are used to today.

MIDI signals are divided into channels; each channel is a discreet signal that can be directed to a specific device or a number of devices. If you generate a MIDI signal on MIDI Channel 1 and send that signal to a group of synthesizers, then any synthesizer programmed to play a note on either MIDI Channel 1 or MIDI Channel Omni will play. Any synthesizer programmed to respond to MIDI Channel 2 through 128 will not play.

Most musicians had more than one synthesizer, so the problem became how to get MIDI data from one controlling device (the sequencer; either a dedicated piece of hardware or a software running on a Personal Computer) to more than one synth. The answer was to daisy chain the synthesizers together, so that each synth could receive MIDI data pertinent to it as well as MIDI data for the next synthesizer in the chain.

Therefore, any MIDI-capable synth could have:

- **MIDI IN** used to receive playable MIDI data
- **MIDI THRU** to receive MIDI data to pass along to the next synth in line
- **MIDI OUT** to send MIDI signals from MIDI THRU (or its own internal sequencer) to other devices Modern digital audio workstations don't necessarily have this problem, since many musicians are making music with no hardware synthesizer whatever.

MIDI is generated by the sequencer (Qtractor in this case) and is passed to each software synthesizer over virtual MIDI ports.

A virtual **MIDI THRU** port is nevertheless available, so in the event that you do require MIDI data in a passthrough configuration, you can use the MIDI THRU port in QJackCtl, via the Connections window in Qtractor The concept of MIDI channels are also equally relevant even without hardware.

If you set up QSynth to generate a bassline for your piece on MIDI channel 1, and WhySynth to play the melody, then you must not set WhySynth to use Channel 1, or else all notes being triggered on QSynth will also be triggered on WhySynth because **they are being triggered by the same MIDI Channel**.

With 128 channels to use, you are free to use 128 soft synths, but unless you are seeking unison, each should have a **distinct MIDI Channel**.

You can set the MIDI Channel in the **Track Properties** of each of your Qtractor tracks.

Using Multiple Sequencers

If a synthesizer has a built-in sequencer, then the only MIDI signals that need to be sent to that device are START and STOP. This is common in external drum machines, which inherently manage their own sequencing so that you can create drumbeats and loops on them using a fairly familiar interface (familiar as long as you've used drum machines, anyway).

On the software side, [Hydrogen](#) is a good example. Hydrogen is a self-contained drum machine for Linux which allows the user to create custom patch sets and which does all of its own sequencing.

Therefore, if you were to use Hydrogen as a stand-alone drum machine, you wouldn't sequence your drum tracks in Qtractor's MIDI editor, but in Hydrogen's sequencer. Since Hydrogen is JACK-aware, sending a START signal to Hydrogen to start the sequence playing and a STOP signal when the song is finished, is all that Qtractor will do; the clock and synchronization of the drum beats with the rest of the sequence is handled by JACK just as it would have been on hardware synthesizers via a MIDI clock.

Working in Qtractor

Editing audio in Qtractor is done directly in the workspace, with a small set of powerful tools. There are three types of audio within the Qtractor workspace:

- *Audio files* - the representation of the actual audio file on your harddrive, which is being referenced by Qtractor because you imported it into your project. It may or may not be actually used in the project yet, but if you have linked to it and so it will appear in the Files panel.
- *Track* - the representation of a mixable channel in Qtractor's workspace. It may or may not have audio in it, but it is a bus which can hold audio, effects (or, technically, MIDI data, but that hardly applies in a chapter on audio editing).
- *Clip* - a piece or the whole of an audio file, as it appears in your Qtractor project. **Audio clips are always contained in an audio track.** You can edit and effect the latter two; the first (the audio file itself) is merely referenced in Qtractor and cannot be directly manipulated by any of Qtractor's tools. In fact, it should be noted that Qtractor, like most other digital audio workstations, is not a waveform editor; for cosmetic corrections such as removing pops or blown-out plosives, or removing lip smacks or dampening roomtone, you should use a waveform editor such as Audacity.

(Not) Editing Waveforms

Cleaning your source sound is frequently one of the first tasks you'll do, so ideally your sound will be sanitized before ever reaching Qtractor. However, in the event that you have imported audio and started working and then you decide that something needs some detail work, you can export just the clip that requires cleaning from Qtractor, clean it, and then re-import it. To export a clip of a sound file for cleaning outside of Qtractor:

1. Right-click on the clip in its track and select *Clip > Export* from the contextual menu.
2. Edit the exported clip in a waveform editor such as Audacity and save export your changes.
3. Import the modified clip back into Qtractor and move it into place over the previous clip.

Audio Clip Tools

Outside of cleaning the sound of an audio clip, there are plenty of modifications you can do to the audio clips you record or import into Qtractor, such as **splitting**, **punching-in**, **looping**, and more.

Moving Clips and Controlling Snapping

Your mouse cursor has four selection modes. Its default mode is *Clip select*, which allows you to click and drag clips within and between tracks. The precision with which you can move clips back and forward in time is determined by snapping. To deactivate or change snapping, click the **View** menu and select *Snapping* to choose the resolution you want to use when moving clips; you can move clips by beats, quarter notes, sixteenth notes, and so on, or none to deactivate snapping.

Truncating (Splitting) and Extending Clips

Also in clip mode, you can split audio clips nondestructively. To split a clip:

1. Position the playhead in the timecode bar at the top of the Qtractor workspace to the position at which you'd like to split your audio clip
2. Click the clip you want to split so that it is selected
3. Click the *Clip* menu and choose *Split*

The clip has now been split at the position of the playhead.

Since Qtractor never affects the original audio file it is referencing, any split you make to an audio clip is reversible. The easiest way to undo a clip (aside from using *Edit > Undo*) is to extend the clip. To extend a clip, click on the edge of the clip with your selection arrow and click and pull the edge to reveal more of the audio file within the track.

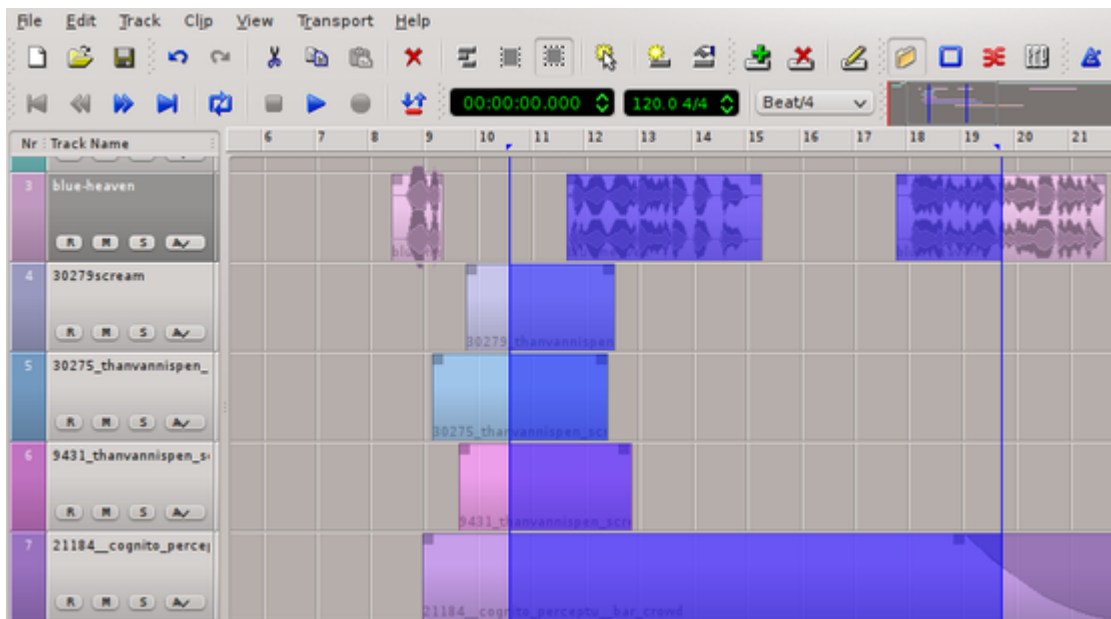
Extending or truncating clips by dragging the clip's edge is also affected by the **Snap** setting, so if you are having difficulty modifying the end or beginning of a clip, check to make snapping is deactivated.

A second way to split clips is to use the *Range* or *Rectangle* selection tools.

Range and Rectangle Selection

The default selection tool is the *Clip* selection, which allows you to click on clips and select them as a whole so that you can move them within the workspace. The other two modes are *Range* and *Rectangle*.

The *Range* selection tool spans all tracks in your workspace and allows you to arbitrarily select any portion of clips regardless of where the clips begin or end. In other words, you can select a range in the very middle of a clip, and all other clips above and below it because you are selecting a block of time, rather than basing your selection on the nature of the clips in your workspace.



To use the *Range* select mode:

1. Click the *Edit menu > Select Mode > Range*
2. Click within an audio clip and drag your selection

Using Range Markers

The *Rectangle* selection mode is similar to *Range* but is track-specific. Using the *Rectangle* selection tool permits you to draw the range of selection over one or more tracks, and a selection will be made wherever a clip is present.

To use the *Rectangle* select mode:

1. Click the *Edit menu > Select Mode > Rectangle*
2. Click within an audio clip and drag your selection. You may select portions, or the whole, of clips on as many tracks as you wish. Or you may confine your selection to one. With both tools, your selected areas can be cut, copied, pasted, deleted, or even lifted out from their clips and moved elsewhere (the splits will be made automatically for you).

To lift a selection from a clip or a set of clips, click and drag the selection.

To copy and paste, click on the *Edit* menu and select *Copy* or *Cut* or the usual keyboard shortcuts of **Ctrl+C** or **Ctrl+X**. When you are ready to paste, click the *Edit* menu again and select *Paste* (or use **Ctrl+V**) and click in the time line where you want to paste the clips.

If you have multiple clips copied., then they will be pasted relative to how they were copied (so if you paste a clip originally from tracks 1 and 2 into track 3, then the pasted clips will fall into tracks 3 and 4).

An alternate way to select a range in your workspace is with the marker transports, which appear in the timeline above the workspace alongside the play transport.

To bring the markers to your cursor

1. Click in the timeline above the Qtractor workspace. Both the *In-* and *Out-* markers will appear at your cursor's position.
2. The position of the markers is the out-point for what will become your selection. Move your cursor left and click again to set the in-point.
3. The space between the two blue markers is now the active range. Click the Edit menu and use the options in the Select submenu to select areas based on the currently active track, the range spanning all tracks, and so on.

You can also use this range to define a loopable area, or to delete or copy and paste its contents.

Paste Repeat, or “Looping”

In electronic music especially, there's a common need to record one or two measures and then loop those measures to create standardized basslines or drumbeats. The “right” way to achieve that would be to use a drum machine or a sequencer, which would literally play the same data (either pure MIDI data or audio samples) for as many measures as the composer programs.

However, the effect can be emulated in Qtractor with a Paste Repeat, which allows you to copy an audio or MIDI clip and then paste it back-to-back for as many repetitions as you want.

To perform a *Paste Repeat*:

1. Check your snapping settings in the View menu (usually when looping clips, the user wants precision snapping to the first clip's edge).
2. Click to select the clip you want to copy (or use a range selection to extract a portion of the clip) and use *Edit > Copy* to Copy the clip.
3. Click *Edit* and choose *Paste Repeat*.
4. In the *Paste Repeat* window, enter the number of iterations you want to paste, or the duration (in timecode, frames, or beats) you need to fill with repetitions. Click the *OK* button to confirm.
5. Click in the timeline to complete the paste and anchor the newly pasted clips in their track.

Punch In/Out

If one or two measures of a track doesn't quite measure up to your standards, you might want to re-record just those measures without having to record the entire track over. This process is called “punching in”.

To perform a punch-in (and punch-out)

- Use the *Range* select tool or the range markers to define a region of your timeline as the active area as described in Section Range and Rectangle Selection and Section Using Range Markers.
- Once your range is set, click the *Punch In/Out* button in the main toolbar, or use the *Transport* menu and select *Punch Set*.

- Arm the recording destination track by clicking the *R* button in the track list.
- Click the *Record* button in the main toolbar, and then the *Play* button. Qtractor will play back your piece, and you can play along but nothing you play will actually be recorded except within the **punch range**. Recording will automatically stop once the transport leaves the punch range.
- To stop playback, click the *Stop* button in the main toolbar, or use the space bar. Punching in and out is a common task used in both MIDI and Audio recording.

Looping Playback

To play a section of your composition in a **loop** (often done so the musician can practise for a punch-in or improvise), select a range as described in Section Range and Rectangle Selection and Section Using Range Markers.

Once your desired loop range is selected:

1. Click on the *Transport* menu and select *Set Loop*.
2. Move the transport into the loop range, or start playing immediately; when the transport reaches the end of the loop range, it will return to the in-point of the loop and seamlessly continue playing until stopped.

Simple Fades

Fading in or out of an audio clip is common enough that there is a quick shortcut to achieve the effect on any clip. Notice in the top corner of any clip in the workspace there is a semi-transparent square node. Clicking and dragging this node further into the clip will create a **fade in** (if done at the beginning of a clip) or a **fade out** (if done at the end of a clip). The fade is given a slight curve to give it a more natural feel.

Merging Clips

Most of the time, these simple fades achieve the effect you need but should you desire finer control over your fades, see Chapter Automation.

Merging Clips

After a long day of editing, your workspace might start to look fragmented, with pieces of audio clips appearing in their tracks with long stretches of empty space. This is not necessarily a bad thing, but it does sometimes allow for accidental moves of precisely timed sound cues and it also may reflect a number of disparate files on your harddrive even though you consider the track one instance of music.

By merging clips, you can change a number of clips into one consolidated file on your harddrive:

1. Select the clips within a track that you want to merge into one.
2. Click the *Clip* menu and select *Merge*.
3. In the *Merge/Export* window, enter a new name for the merged file and save it to a logical location on your harddrive.

4. The merged audio clip immediately replaces the old files in your project.

Track Edits

As audio clips contain the sound (and MIDI) files you are using in your Qtractor project, so do Tracks contain the clips themselves. Tracks allow you to mix the audio clips in relation to one another, apply effects to whole groups of clips without ever affecting the waveform files themselves (ie, nondestructively), and even automate everything from volume to panning to the functions of effect units and synths.

A special window is dedicated to managing tracks: the *Mixer* window ; set up like a traditional mixing desk with a user-friendly interface to allow for *Aux Sends*, *Returns*, *Sub-mixing*, and more. You can also manage many aspects of your tracks through the track list on the left of the Qtractor interface, but since the Mixing desk interface is so ubiquitous and useful, it's common to have it open on a separate monitor or virtual desktop for quick reference.

To open the *Mixer* window, click on the *View* menu, select *Windows*, and choose *Mixer*.

Most track functions can be done through both the main Qtractor interface, as well as through the **Mixer** window.

Track Order, Height, and Properties

It's a good idea to keep related musical parts grouped together. If your drums appear on track 1 and your bass on track 14, it's difficult to edit the two tracks in unison as one might reasonably attempt to do.

To keep a logical order to your track listing, you can rearrange where each track appears in the Qtractor workspace.

To move a track up or down the track list, either click and drag the number on the far left of the track label, or use the *Track* menu and select *Move* to move the currently highlighted track either up, down, or straight to the top or bottom of the list.

In addition to changing track order, you can change the height of tracks with the Height selection in the *Track* menu. Other track properties, such as track name, the color of the clips, the type (audio or midi) of track data, and more, can be modified through the *Track* menu, especially the *Track Properties* selection.

Mixing

Mixing in Qtractor, like most digital audio workstation software, is modeled after mixing on real-world equipment. With Qtractor, as with hardware mixing equipment, you are able to define inputs, outputs, auxiliary sends and returns, effects, and automation.

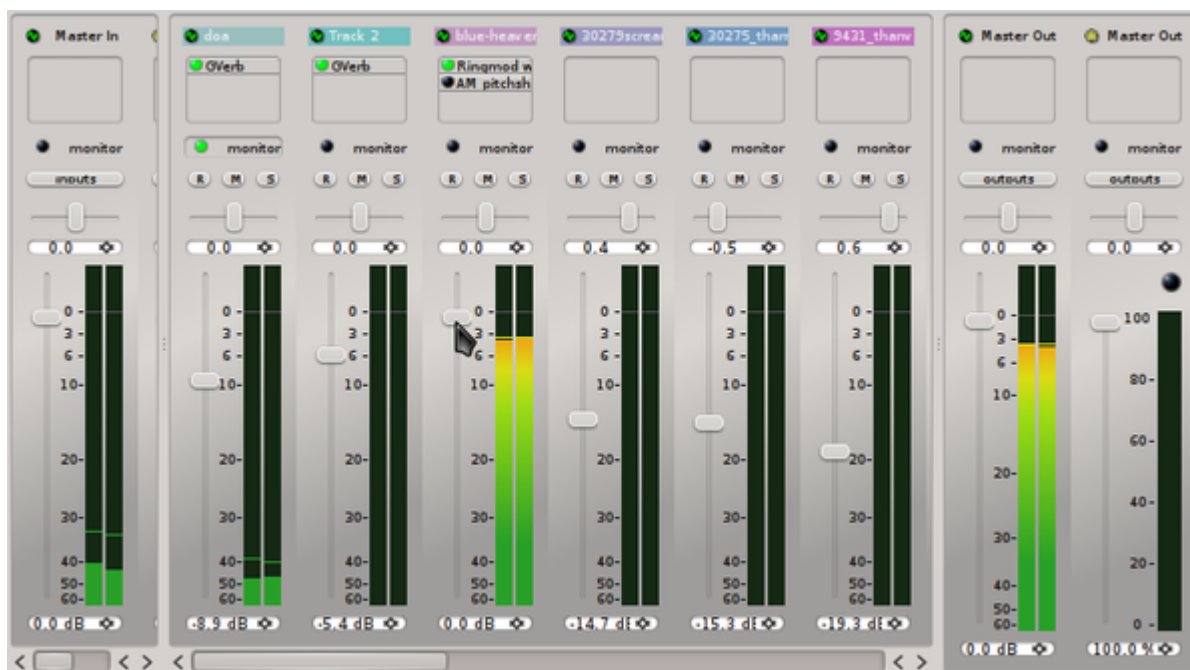
Generally, mixing in Qtractor utilizes three different toolsets:

- **Mixer** - used to control the stereo separation and levels of the track, route sound, and configure effects and other outboard “gear”
- **Effect Units and Filters** - plugins used to sweeten, degrade, fix, or control sound
- **Automation** - makes Qtractor your live mixer so that levels, effects, and pans are adjusted during playback Just as with a hardware-based studio, you'll use a combination of

techniques to achieve the mood, sound, and quality you want. The central hub, however, is the *Mixer* window.

Mixer

The Mixer window can be accessed by clicking the *View* menu and selecting *Mixer* from the *Windows* category. The Mixer opens in its own window, making it a prime candidate for occupying a second screen if you have a multiple-head setup.



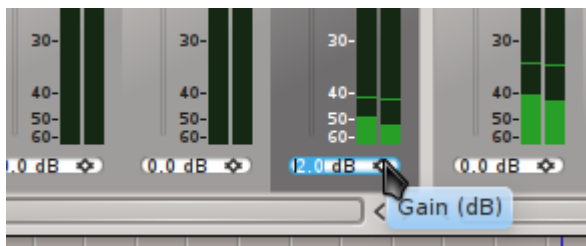
On the extremities of the Mixer window are the *Master In* and *Master Out* controls. These are independent of any one track and control everything coming into or going out of the mixing board. You can adjust the master levels of the input and output, and place any effect processors you want to apply globally (such as a compressor to control peaks on an outboard microphone, or to prevent peaks on outgoing audio).

The middle portion of the Mixer window represents the tracks in your project, both **audio** and **MIDI** (distinguished by the *Audio* or *Midi* icon next to the name of the track).

These control the individual track, allowing you to set the level of the track with the volume slider and gain control, the stereo position with the **pan/pot** control, and **mute**, **solo**, or **arm** the track for recording.

Setting Levels

When monitoring the levels of your tracks, make sure that your sound does not enter the red zone on the volumeter. In audio production, any sound that goes above 0 dB hits its peak and will cause distortion. It's not uncommon to flirt with 0 dB to truly maximize the impact of important sounds, but leaving sounds above 0dB in your final mix is a surefire way to ruin your project. Using the volume level slider, adjust your track volumes into safe settings while listening to your project. Then listen to your project again to refine the levels in relation to one another. If the sound source is too low (or too loud), you can add (or subtract) gain at the input point with the Gain adjustment at the very bottom of the mixer window.



This adds or subtracts gain level at the point of the sound's entry, meaning that you are changing the amount of sound reaching any effects processors. Generally, gain adjustment is avoided and other tools, like a Compressor or Equalizer, are favoured.

Tracks levels don't exist in a vacuum; the levels are cumulative...

So if you have two tracks with near-maximum levels that produce very loud sounds at the same time, the *Master Out* level will **peak** even if each track was "safe" independently.

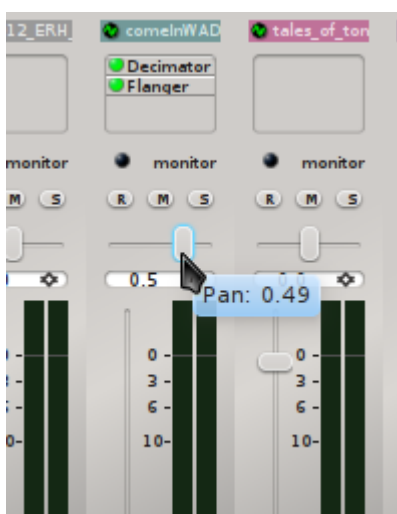
There is no authoritative way of achieving a balanced mix ; some professional sound mixers prefer an additive mix, in which they start all levels at a reasonable and safe level and then accentuate important parts by adding volume to tracks.

Others prefer a subtractive method by starting everything at the highest safe (or unsafe) level and then bringing volumes down so that everything falls into place. Predictably, still others prefer a combination of these methods.

Stereo Separation

By default, tracks are set to play sounds from the center point of the stereo space, meaning that to the listener the sound will feel as if it's situated directly in front of them, or evenly between their left and right ear.

To diversify the sound, sometimes in order to emulate a live setup or to suggest that a sound is just next to the listener, or to broaden the scope and impact of the piece, you can **pan** sound between the left and right speakers.



When panning a track, "less" is often "more" and an even spread is usually best. In other words, do not arbitrarily assign tracks in stereo space just to give a spread to your sound; consider where the sound would realistically come from if the listener were watching a live performance.

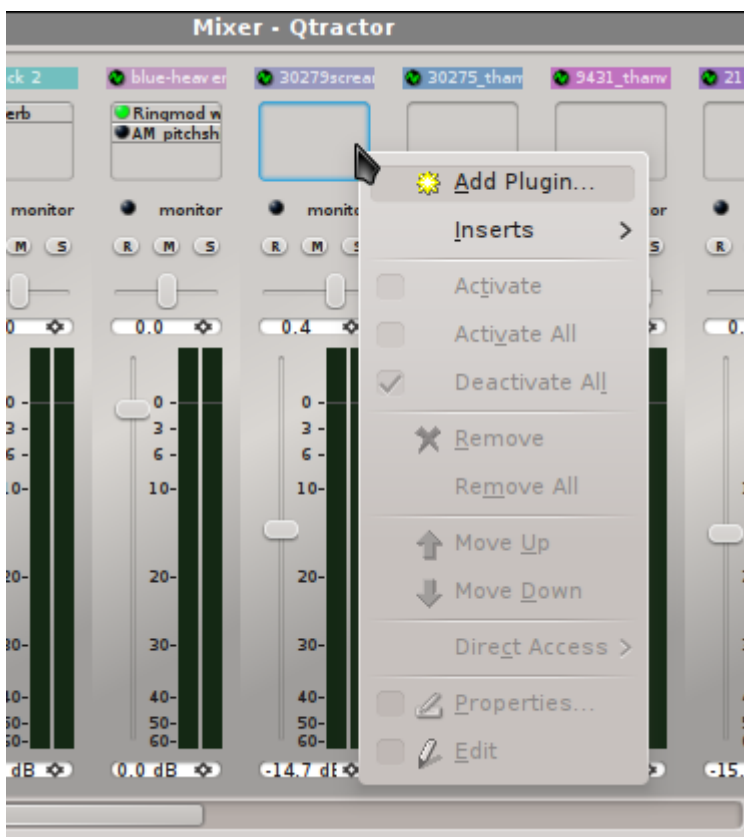
Avoid moving sounds to the extreme stereo positions, since in real life sounds are rarely only heard in one ear and not the other (the sound of extreme stereo separation was common in early stereo mixes from the 1960s, in which the drums might only be heard in the right ear and a guitar only in the left; the effect was revolutionary at the time but largely considered utterly unnatural now).

A gentle spread of sound is most natural, with sounds at the extreme edges of the stereo space being used for atmosphere, emphasis, or effect.

Plugins

In the *Mixer* window, effects processor can be added to any track (the *Master In*, *Out*, or individual tracks in the project):

- Right-click the plugin window in the track into which you wish to add the effect or synth.
- Select *Add Plugin* from the contextual menu.



- From the *Plugins* window, choose the **LADSPA**, **LV2**, **DSSI**, or **VST** plugin you want to plug into the track. Click *OK* to continue. A plugin may also be a MIDI soft synth that you wish to associate with a track of MIDI data.

The order of processors is important ; applying a reverb effect first and then a distortion filter will render a track with distorted reverb. Applying the distortion filter first and then the reverb will render a track with reverberating distortion.

To move an effect up or down the order of processing (ie, to place a reverb effect before a distortion filter), right-click the effect you want to move and select *Move Up* or *Move Down* from the contextual menu.

Once a plugin has been inserted into a track, control its settings in the pop up window that appears for the plugin. After your initial adjustments are made, you can access a plugin's controls at any time by right-clicking on the name of the effect unit in the track and selecting *Properties* from the contextual menu.

A green light to the left of the plugin name indicates that the plugin is active. You can temporarily deactivate (**bypass**) the effect by right-clicking the effect in the track and toggling off the *Activate* menu item. If you wish to by-pass all effect units in a track, choose *Deactivate All*. If you want to remove an effect or plugin from a track entirely, right-click on the effect in the track and choose *Remove* or *Remove All* to remove all plugins from that track.

Send and Returns

What we call a “plug-in” in a DAW would be most akin, in a hardware-based studio, to a built-in reverb effect or gain adjustment on some of the more general-purpose mixing desks on the market. Mostly, however, there are no “plug-ins” in a traditional studio setup; everything is outboard and the mixer plugs into the effect unit, the effect unit processes the sound, and then the affected sound is send back into the board to be integrated into the mix.

This model is still useful in the software world, since not all sound applications are available as plugins, or you prefer to run them as separate applications. To be able to utilize Sends and Returns, you should have some external, JACK-aware, sound application installed. In other words, the application should run outside of Qtractor but be programmed to utilize JACK when available. A few examples of such applications would be

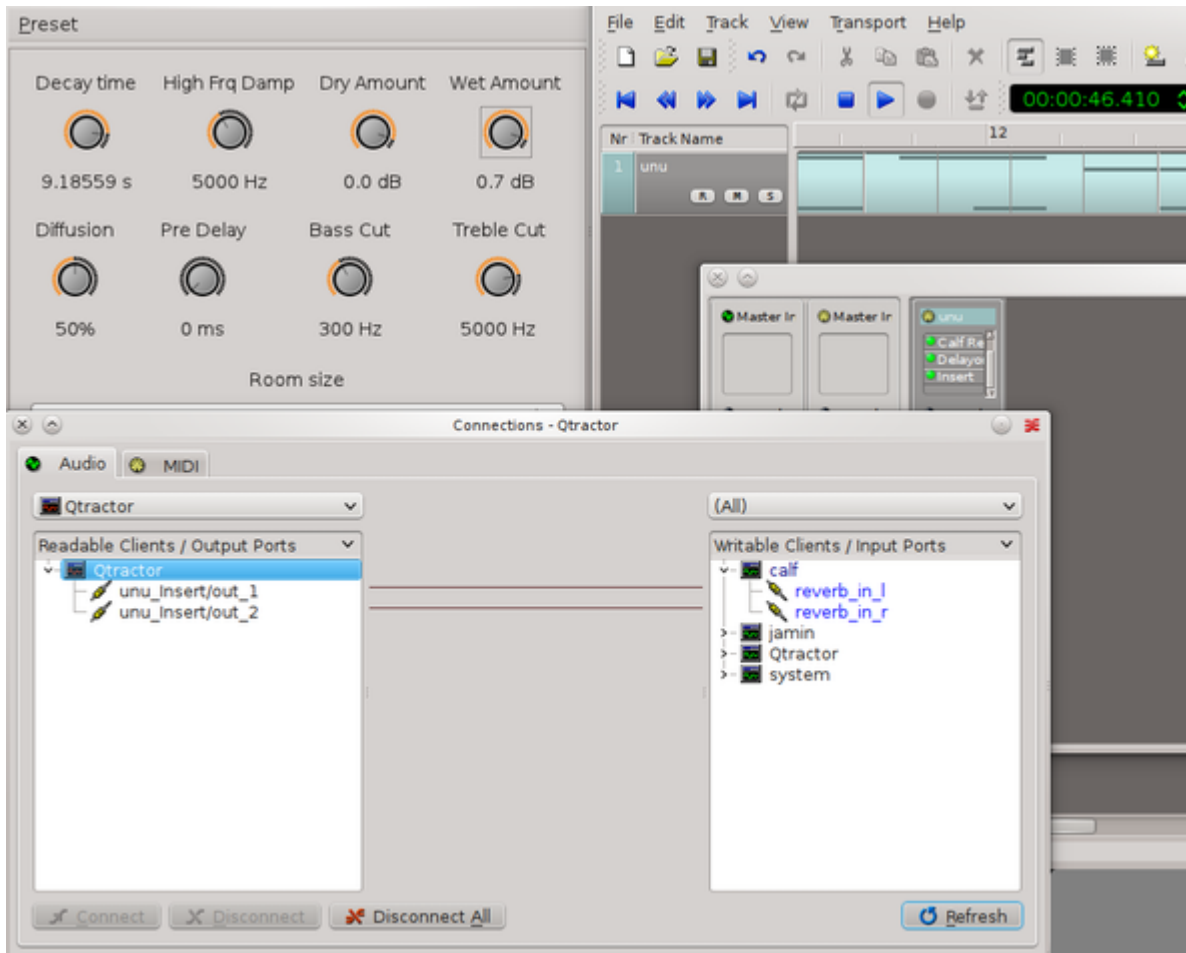
- Jamin - a sound mastering application
- CALF - an audio plugin set that is available as a stand-alone application as well as a plugin
- [amSynth](#) - a soft synth
- QSynth - a fluidsynth frontend
- [Schismtracker](#) - a MIDI tracker
- [Hydrogen](#) - a stand-alone drum machine

For this example, launch the CALF JACK Host plugin client. Click the *Add plugin* button and select *Reverb* and turn the *Wet Amount* nearly all the way up so that it will be easy to hear that the sound is being effected in this test. Bring in or generate sound into a track in Qtractor. To create a *Send* and *Return* on that track:

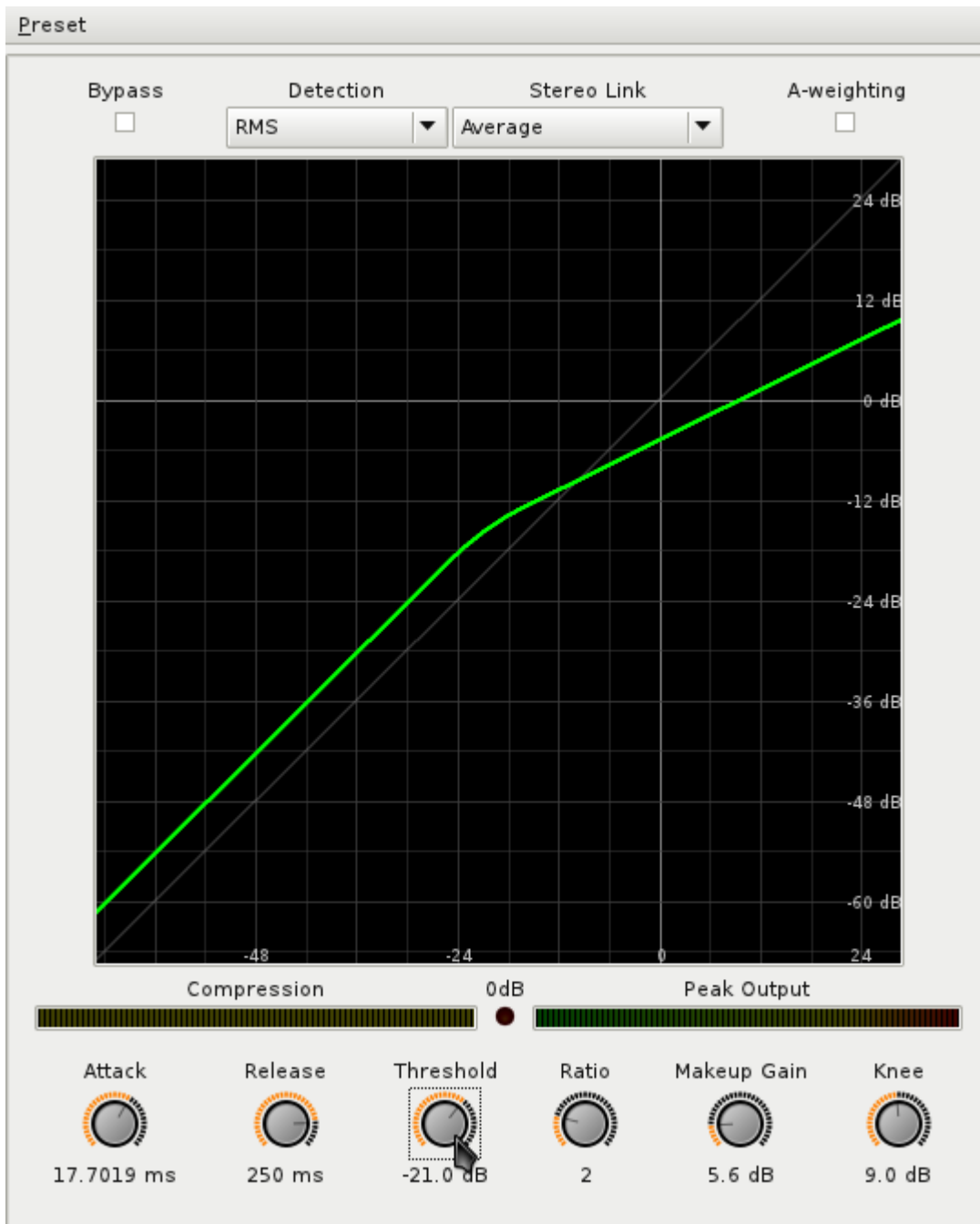
1. Locate the track you wish to affect in the *Mixer* window in Qtractor.
2. Right-click the plugin inset and choose *Inserts*.
3. In the *Insert* window, click the *Sends* button in the lower left corner.
4. In the *Connections* window that appears, choose the *Insert Out* connection in the left column and then then *reverb in* of *calf in* in the right column. Click the *Connect* button to route the sound. **Repeat this for both left and right channels.**
5. At this point, you are successfully sending the sound from your track to the CALF effect processor, but if you were to play the track in Qtractor you wouldn't hear anything. Of

course, the reason is because there is no *Return* yet. Back in the *Insert* window, click the *Returns* button.

6. In the *Connections* window, choose the *reverb out* connection in the left column and then the *insert in* of Qtractor in the right column. Click the *Connect* button to route the sound. **Repeat this for both left and right channels.**
7. Now your track is routed out to the CALF effect unit, processed, and sent back into Qtractor. Click the *Play* button in Qtractor to hear the results.



Effect Processors and Filters



There are many theories, personal styles, and preferences when it comes to mixing a project. One producer might opt for no effects at all while another might use compressors, limiters, levellers, EQ, and more, and those same producers will abandon their style depending on what band they are dealing with.

The best way to know what effects to use is to experiment ; discover what different filters do, find out which ones you like or dislike, and do whatever your ear tells you to do. There are some staples in any audio engineer's toolkit, and between the [Steve Harris LADSPA collection](#), [CALF](#), and [Jamin](#), you'll find everything you need to achieve the sound you want:

Compressors

As a sys admin has ssh and emacs, an audio producer has Dynamic Range Compressors. A compressor's job is to reduce loud sounds or to increase quiet sounds by reducing (or "compressing") the signal's overall dynamic range. It has a number of uses in music production.

A compressor on a track that is inconsistent in dynamics will produce a more consistent track that might be easier to mix into the rest of the piece, and also provide a better listening experience for the audience; it's difficult to follow the lyrics of a song, for example, when the voice track continually drops behind the other instruments.

Compressors are very common on drums, as well, since when drums are mixed the loudest hits and crashes tend to rise to the surface while all the subtleties, like little touches of ornamentation and the decays of drum and cymbal hits, are lost. A compressor will even out the range between those softer sounds and the harder sounds, so you still get the hard-hitting beats but get to keep the personality.

There are at least three good compressors for GNU Linux:

CALF

The Calf plugin suite contains a compressor which you can use either as a plugin from within Qtractor or as a stand-alone app as an Aux Send/Return. Its features are typical of an all-purpose, general-use compressor, and running its full GUI with caljackhost might help you understand the theory behind compression if you are new to it.

If you are new to compressors, try this experiment:

- Open the CALF compressor by launching `caljackhost`. Click on the *Add plugin* menu and choose *Compressor*.
- To launch the user interface, click the *Compressor* button in the lower left. Change the settings to:
 - Ratio to 1
 - Makeup gain to 0dB
 - Threshold to -43.2
 - Knee to 0dB

Notice that this causes the green line to align perfectly with the gray line that runs diagonally through the compressor screen. This achieves, essentially, a pass-through effect. You have set a limit for every possible level of sound equal to itself, and since no possible level of sound will ever be greater than itself, nothing will be constrained.

- Now change the ratio to approximately 2 (a 2:1 ratio). Notice the green line now deviates from the gray normal line; you may think of the green line as a sort of ceiling. Measuring soundwaves by the numbers along the X-axis, you can see that 24dB in the input (ie, the gray line), it will be constrained down to 0dB (along the green line). Accordingly, a soundwave that reaches -12dB will be constrained to -24db.
- For one last demonstration, turn the *Ratio* knob all the way to right so it reads +inf and the *Threshold* knob all the way to the right so it reads 0dB. This places a hard ceiling at the

current threshold setting. This means that any sound louder than the threshold will be constrained to that level, no matter what. Set too severely this can cause a clipped sound, but the concept of leveling is ubiquitous in production, as a safeguard to ensure that the levels do not go over 0dB.

- When exactly compression kicks in is controlled by the *Threshold* value. Try adjusting the threshold knob and notice that the bend (or “**knee**”) in the green line changes. Turn the *Ratio* knob back to 2 and set the *Threshold* to -12dB. Setting the threshold to -12dB means that everything below -12dB will remain unchanged, while anything above -12dB will be constrained to some degree (depending on the ratio setting, of course).
- Make the Knee more gradual and natural by adjusting the knee to 18dB; this gives a softer curve to the compression. Sometimes this is intentionally not increased, and a sharper knee is desired (it’s a trick sometimes used in compressing drums, to create hits that “pop”).
- The *Makeup Gain* knob, currently set to 0, allows you to boost the output sound to account for the inherent loss in volume when using a compressor that constrains levels.
- The *Attack* and *Release* knobs control how quickly the compression effect takes to take full effect, and how quickly the effect is released. In other words, at a 0ms attack, a soundwave will be constrained the very moment it pushes toward the threshold setting. The result of this is often a very harsh and noticeable effect, usually undesirable (except, again, for some stylistic choice sometimes made on drum kits or special effects). Setting the attack to 500ms (.5 second), for instance, renders a softer, more gradual effect that better allows a soundwave to retain its natural shape and curve but still suppressing it under the threshold’s level.

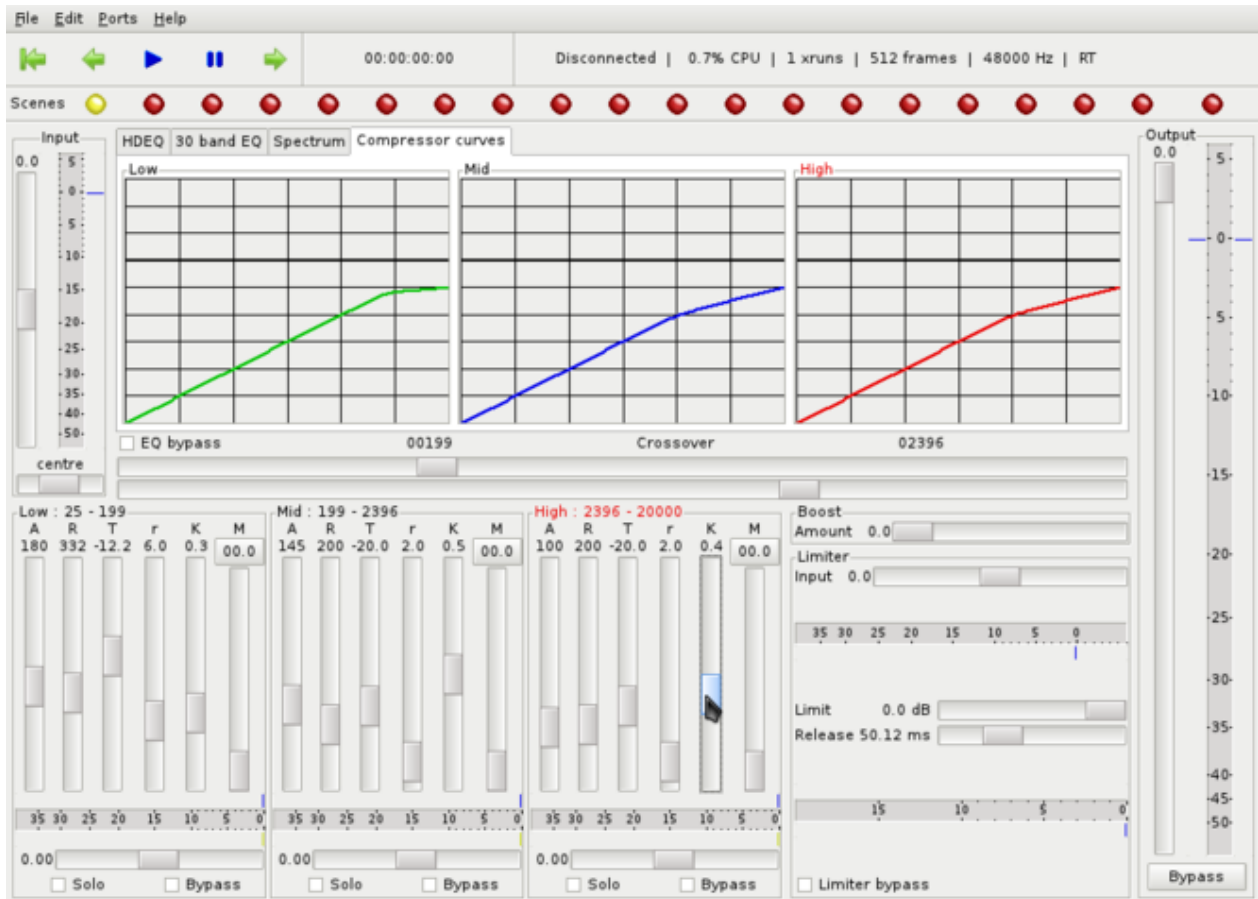
Once you’ve designed good compression settings, **save it as a preset** by clicking on the *Preset* menu and choosing *Store preset selection*. You will need the preset saved (perhaps in your Qtractor project folder for better organization) since Qtractor obviously will not save or preserve your settings. The Calf compressor is a capable and user-friendly compressor unit, available either as a stand-alone unit or a plugin from within Qtractor, so use it often!

Steve Harris LADSPA Dyson Compressor

The Dyson Compressor is included in [the Steve Harris set of LADSPA plugins](#). It is very basic, with few adjustable settings, but if quick and basic compression is all you need then it may just be what you need.

Use the Dyson compressor as you would any other LADSPA plugin in Qtractor. Its interface offers four sliders for customization: the peak limit (ie, threshold), release time (in seconds), the fast compression ratio, and the compression ratio (both ratios are counted from 0 to 1, so some math is required, with 1:1 ratio being equal to 0).

Jamin



The [Jamin](#) mastering application contains one of the most powerful compressors available. Whereas the Calf compressor excels in its simplicity, Jamin provides three separate compressors, essentially, for the Low, Mid, and Hi frequencies of your input.

This obviously allows for a far greater degree of fine-tuning. As for what Jamin defines as a low, mid, or hi frequency: even these are adjustable with the middle horizontal Crossover sliders. Your familiar Attack, Release, Threshold, Ratio, Knee, and Makeup Gain controls are found at the bottom of the window in the form of vertical sliders.

As with the stand-alone version of Calf, Jamin is used as an Aux Send / Return; unlike Calf, it has no plugin option.

Limiters

Because Jamin also features a full-featured 30-band EQ, it might be best treated as the compressor/ EQ filter for the Qtractor's Master Out; something that Jamin obviously intends since it is advertised as a [mastering](#) application. As with the stand-alone version of Calf compressor, you can and should save your presets in Jamin by clicking on the *File* menu and selecting *Save As*.

Directly related to dynamic compressors are limiters, which are basically **compressors set to an infinite compression ratio**. There are limiters available in the Steve Harris LADSPA set. The Hard Limiter, for example, enforces the threshold by literally clipping off the offending portion of the sound wave. Its upper dB limit is 0dB and the harshness of the clipping can be set with the Wet Level (the more wet, the more clipping) and Residue level, which essentially control the attack and release times of the effect.

Limiters, compared to compressors, are by nature fairly harsh. However, slightly clipping the upper tips of offending soundwaves can create the illusion of loudness without actually crossing over into true distortion levels. It may also be used as a protective filter on the *Master Out*, just to ensure that levels absolutely do not rise above 0dB.

Normalization

The idea of normalization basically refers to dynamic range compression, since its job is to take irregular sound and make it more “normal”. However, the term is imprecise and so it can mean many things.

In Qtractor, tracks that are too soft can have gain applied; manually applying gain can be dangerous, since you do not know how much gain you can safely apply without sending peaks over the 0dB mark. To have Qtractor calculate a safe clip-wide gain level:

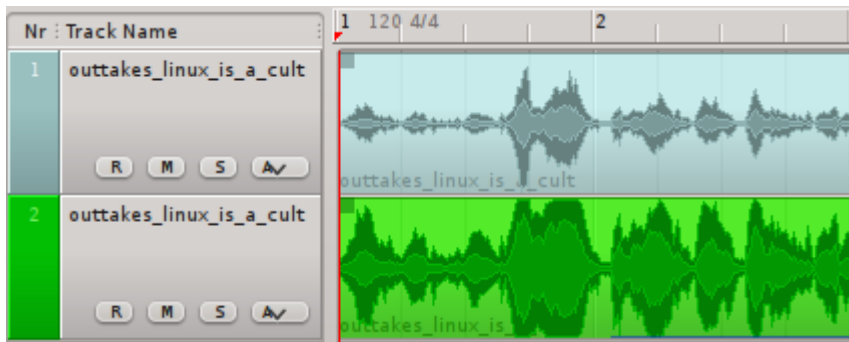
1. Select the clip that is too soft
2. Click the *Clip* menu and select *Normalize*

The clip has a safe gain level nondestructively applied.

What this does not do is compress the dynamic range of that clip. If you want to normalize a clip with dynamic range compression, then use either a compressor such as Calf or Dyson, or process the clip externally of Qtractor. A good external normalization application is *normalize* (named *normalize-audio* in Debian), a command line application for GNU Linux that is available in most repositories and pre-installed on Slackware. The syntax is simple:

```
normalize -a -6db -l -3db nameOfFile.wav
```

In the above command, *-a* is **amplitude** and *-l* is **limiter**; so the command sets the average amplitude of the sound clip to -6dB and ensures that peaks do not exceed -3dB. Obviously these settings would need adjusting depending on what the sound file contained, but it’s a marvelously efficient command for quick normalization.



In addition to normalizing a clip to itself, *normalize* can normalize a group of files in relation to one another, set thresholds, and much more.

EQ

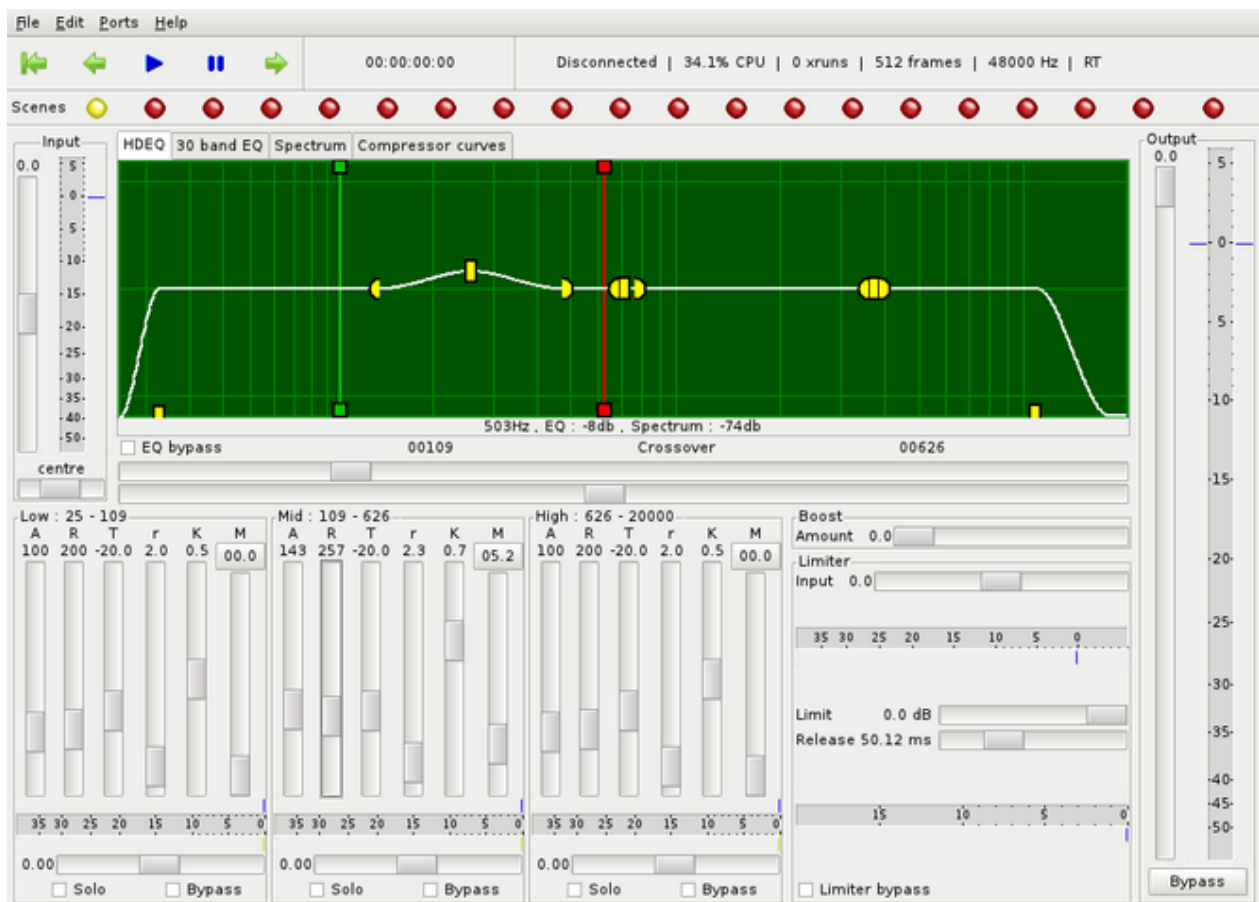
Sound waves, being oscillations of pressure, are measured by the cyclic standard of hertz. The human ear generally can hear from 20Hz to 16,000Hz (or 16kHz), with the human voice usually appearing around the 100Hz to 400Hz range.

A good equalizer can increase or decrease the energy of these frequency bands; this is used by audio engineers to boost bass, or possibly cut bass when it is overpowering, or boost the midtones where the human voice dwells, and so on.

There are a few good equalizers that can be used with Qtractor:

Jamin

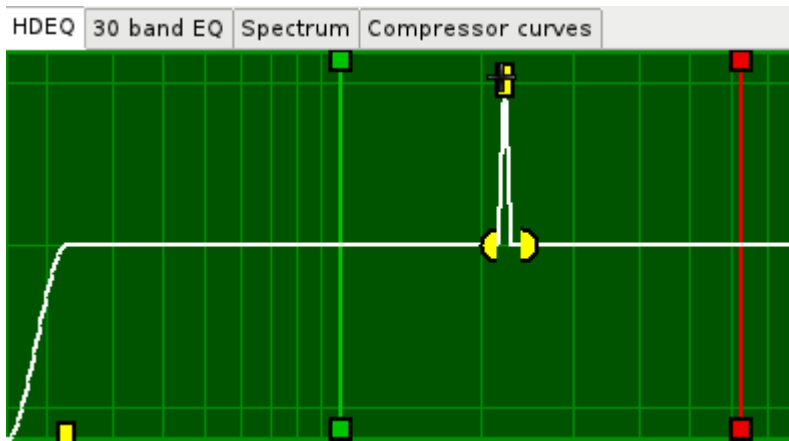
The Jamin mastering application, which you can use as an Aux Send/Return, has two views of EQ: the graphical curve-based HDEQ, and a slider-based 30 band EQ. They are literally two different views of the same thing, so it's typical to use both views depending on what kind of adjustment you need to make and which one makes that adjustment easiest.



The practical theory behind EQ is simple enough; identify the frequency you wish to emphasize or suppress, and then raise or lower its intensity in the equalizer. If you are unfamiliar with equalizers, you may have a difficult time at first being able to identify where in the spectrum of frequencies the sound lies. Put simply, the lower (bass) bands are on the left and the higher (treble) are on the right, with the mids (human voice and similarly pitched instruments) are in the middle.

For more precision, there is an old engineering trick that you can use in Jamin to identify where something you are hearing actually appears in the spectrum:

- Send a track to Jamin as an *Aux Send/Return*
- Open the *HDEQ* view of Jamin and use the yellow curve nodes to create the sharpest curve possible at its maximum value



- Using the middle node, drag the peak across the spectrum, listening to the sounds it emphasizes. Because of the drastic curve, the change will be obvious.
- Once you find the sound you are searching for, lessen the volume of the curve and make it more gradual. Less is more in EQ; rarely is there a time when huge increases or decreases of the existing frequencies are required. If you find yourself wanting to maximize the intensity of a frequency band then quite possibly you need to re-record to achieve the actual sound you were intending to capture.

The other view of the EQ in Jamin is the 30 band EQ tab, which displays the same information as the HDEQ interface, but uses sliders instead of graphical curves and nodes. The same principles apply; the sliders provide a quicker way to boost or cut frequencies when you know exactly what needs to be done and don't want to have to draw curves and move nodes around to achieve it.

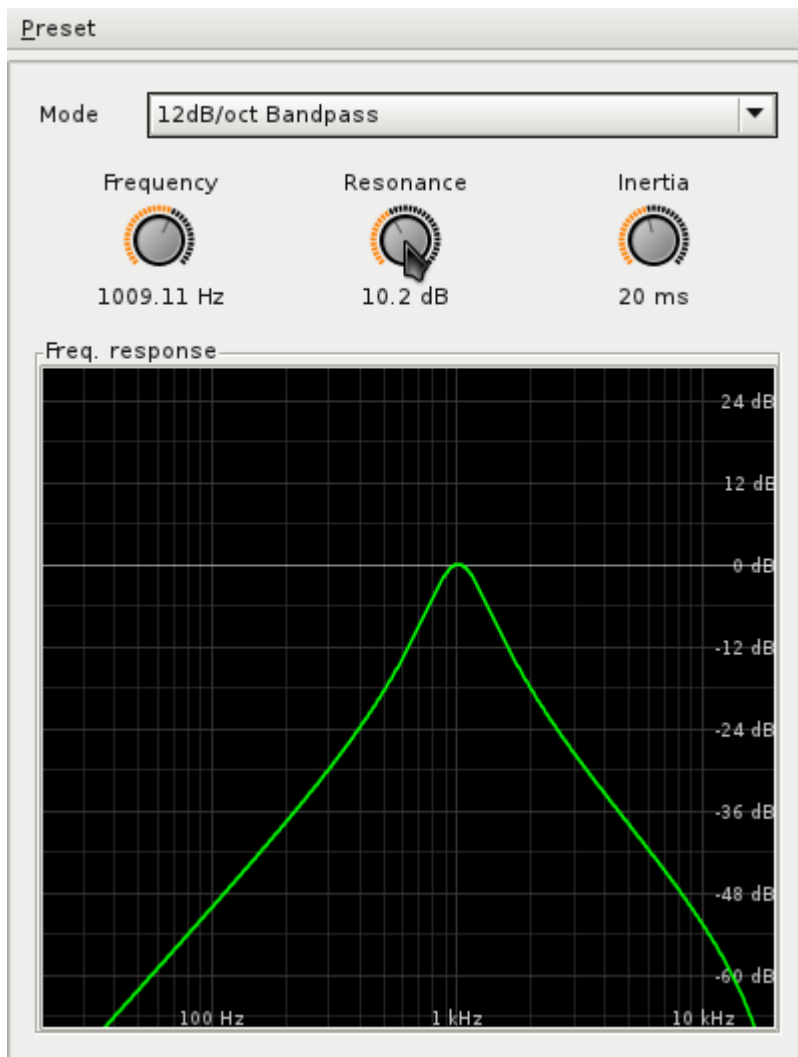
Calf

The Calf plugin set provides a set of filters, which perform EQ through customizable presets. It is less precise than a 30 band equalizer but realistically is often all you'll need. Calf offers four types of presets:

- Lowpass - low pass filters allow low frequencies to pass and cuts off high frequencies
- Highpass - high pass filters allow high frequencies to pass and cuts off low frequencies
- Band Pass - band pass combines low pass and high pass filters configured such that a targeted frequency range is allowed to pass while frequencies above or below that range are rejected
- Band Reject - band reject combines low pass and high pass filters with their intersection being a range of frequencies that will be rejected You have control over the characteristics of these presets via the Frequency, Resonance, and Inertia knobs.

Open the CALF filter plugin by launching caljackhost. Click on the Add plugin menu and choose *Filter*.

To launch the interface, click the Filter button in the lower left corner. When the interface for filters launches, it may only display a list of the presets, so be sure to expand the window until you can see the knobs and filter graph.



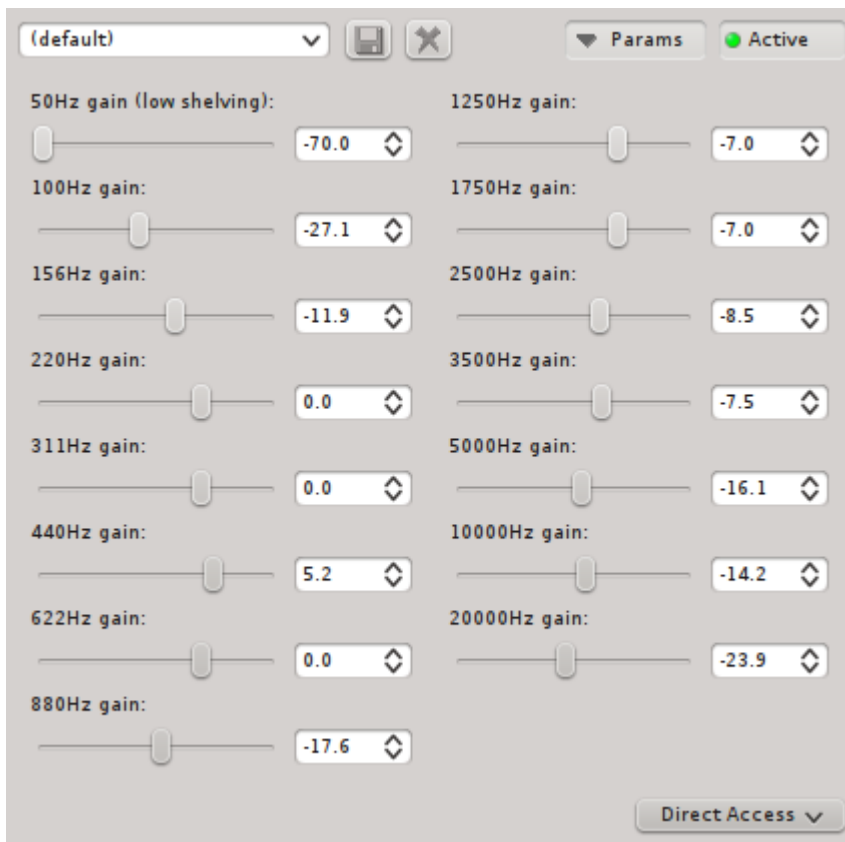
- For the clearest example of how the Calf filters work, select the 18dB Bandpass, Set the *Frequency* knob to approximately 300 Hz. Notice how the filter shape moves horizontally along the graph as you adjust the filter. The green line indicates that sounds at 300 Hz will passthrough the filter, while the frequencies lower and higher than 300 Hz will be filtered down gradually until they are excluded entirely.
- To control what frequencies are attenuated by the filter, use the *Resonance* knob. Setting it to 30dB, for instance, makes for a narrow filter, while setting it to 0dB makes it fairly broad.
- Try a lowpass filter on bass parts, and a highpass on other instruments, and make adjustments by ear.

Once you've designed good EQ settings for your track, save it as a preset by clicking on the *Preset* menu and choosing *Store preset selection*. You will need the preset saved (perhaps in your Qtractor project folder for better organization) since Qtractor obviously will not save or preserve your settings.

The Calf filters are more specialized than the Jamin EQ but just as effective in what it does. Since Calf can be used as either a stand-alone application or as Qtractor plug-ins, they are flexible in how you choose to implement them.

Steve Harris LADSPA Multiband EQ

Simpler than Jamin or Calf is the Steve Harris LADSPA plugin Multiband EQ. It features 15 sliders for broad ranges of frequencies.



There is less customization available but well-rounded curves can still be achieved by staggering the sliders, a common practise on hardware equalizers.

Steve Harris LADSPA DJ EQ

A basic 3-band equalizer, the Steve Harris JD EQ LADSPA plugin allows you to easily control the three general areas of sound frequency: bass, mids, and treble.

Noise Removal

Noise removal is something of a myth borne of automatic filters bundled with recording applications. It is not possible, in spite of what the advertisements say, to isolate noise on a track and eliminate it without also removing that same frequency from what you deem to be signal. The manual elimination of noise is easiest with Jamin HDEQ; first scan your sound for the frequency containing most of the noise as described in Section Jamin, and then invert the resonance curve to eliminate the noise you've found.

Automated noise removal filters do the same process; analyze a sample of user-defined "noise" and then EQ the offending frequencies out.

Reverb

Reverb is an old ever-popular recording effect that simulates the phenomenon of a sound wave bouncing off of surfaces, causing the soundwave to bounce back in a new direction; our ears hear the original sound wave as the source and the reverberated sound wave as an echo.

Common examples are large cathedrals, famous for very saturated and immediate echoes, and canyons, with the stereotypical distant and delayed echo (“Hello....hello.....hello...”). You might also notice less drastic examples in, for instance, small empty closets, or you might try speaking into a coffee tin; the reverberation is immediate and subtle, but it’s still clear that there are soundwaves bouncing back at you.

Reverb effect units simulate this phenomenon and allow the user to control different aspects of how the reverberation is generated.

Calf

As with the other Calf plugins, the Calf reverb can be used as a stand-alone application via caljackhost or as a plugin from within Qtractor. It features a good number of the most important reverb controls:

- *Decay Time* - controls the length that the echo (tail) is heard before
- *High Freq Damp* - *the tail of the reverb is dampened to simulate that sound loses energy and fades out over time and distance. The /High Frequency Dampening knob determines the frequency at which the dampening begins*
- *Dry Amount* - determines how much of the original sound is heard along with the reverb
- *Wet Amount* - determines how much of the reverb is heard
- *Diffusion* - structures the “spread” of the reverb; a highly diffused reverb effect spreads reverb widely across the stereo space, where as a less diffuse reverb effect is more centrally located
- *Pre Delay* - determines the distance between the source signal and the start of the reverb
- *Bass Cut* - determines the frequency at which bass frequencies are truncated
- *Treble Cut* - determines the frequency at which treble frequencies are truncated
- *Room Size* - sets the emulated space, determining the shape and character of the reverberated sound

If you are using Calf as a stand-alone effect unit, remember that once you’ve designed good reverb settings for your track, **you must save it as a preset...**

You can save presets by clicking on the *Preset* menu and choosing *Store preset selection*. You will need the preset saved (perhaps in your Qtractor project folder for better organization) since Qtractor obviously will not save or preserve your settings.

Steve Harris LADSPA GVerb

There are reverb units available in the Steve Harris LADSPA set. The GVerb plugin is particularly powerful.

- *Roomsize* - controls the size of the space being used to simulate reverb
- *Reverb time* - controls decay of the reverb; the longer the reverb time, the longer the “echo” is heard
- *Damping* - the greater the damping, the shorter and weaker the reverb effect will be
- *Input bandwidth* - the strength of the signal that reaches the reverb effect unit for processing
- *Dry signal level* - the volume of the unprocessed source sound
- *Early reflection level* - strength of the initial reverberated wave; a weaker early reflection level has the effect of hearing just the tail of a reverb, while a greater level brings more immediate sound
- *Tail level* - the level of the reverberated sound; increasing this will great a stronger signal, which will therefore continue even past the Reverb time setting, which can then also be controlled by the Damping setting

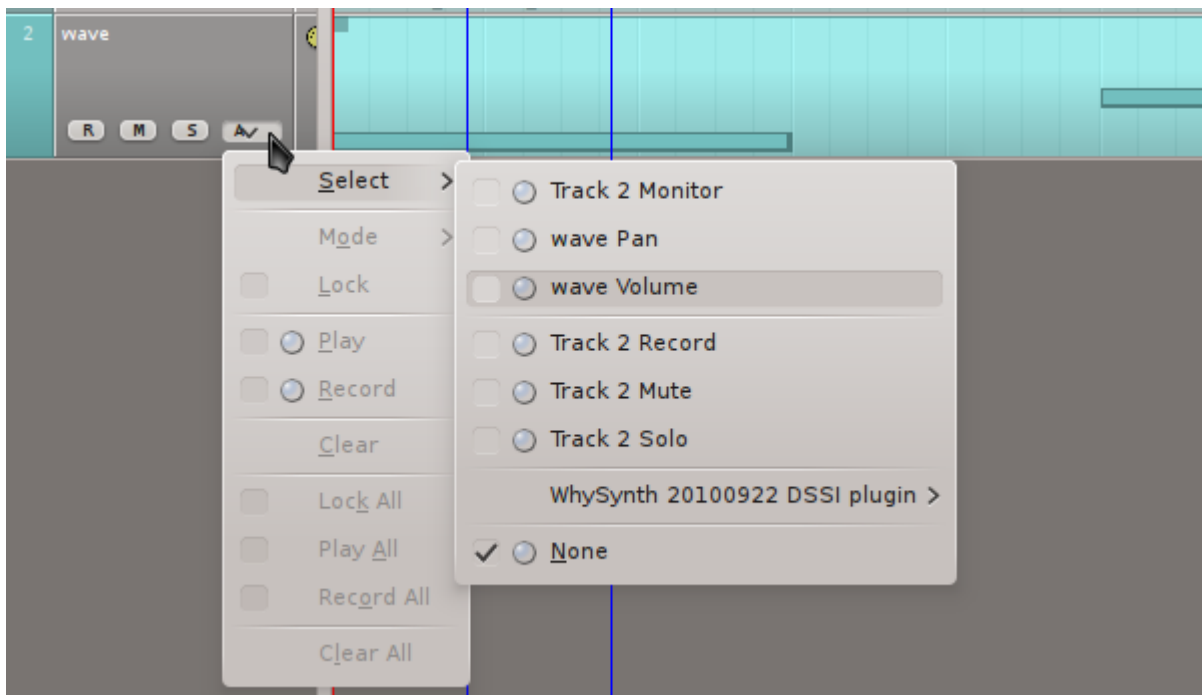
Echos, Delays, Distortion, and More

The range of effect units available for processing sound is nearly endless. With echos, tape delays, distortion filters, amps, decimators, chorus effects, ring modulators, and much more, Linux has no shortage of special effects. Explore Calf and Steve Harris LADSPA plugins to discover a wide variety of new possible sounds.

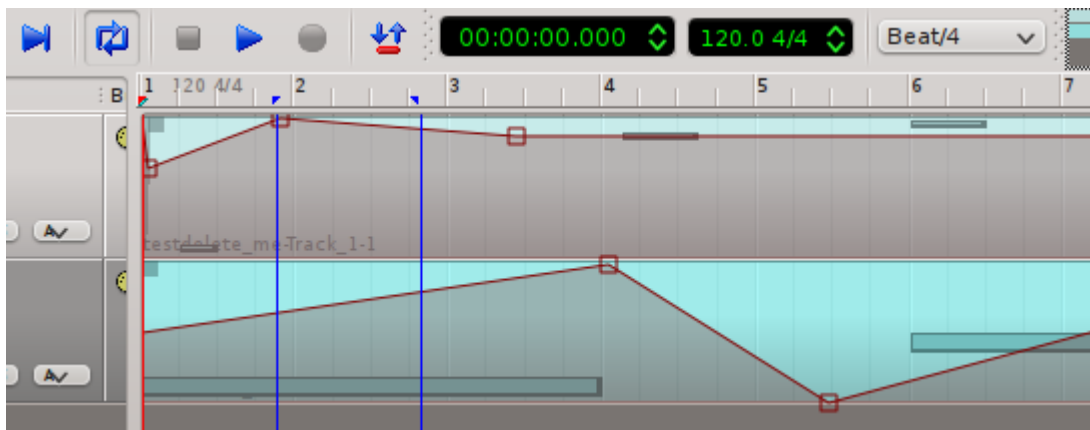
Automation

If you had to mix all of the tracks in a project, at the same time, in realtime, you’d not only need a lot of practice but probably two more arms and another mouse or two. Qtractor features automation of nearly any aspect of any plugin, as well as basic track functions like volume and panning. It is only possible to automate tracks; buses cannot be automated.

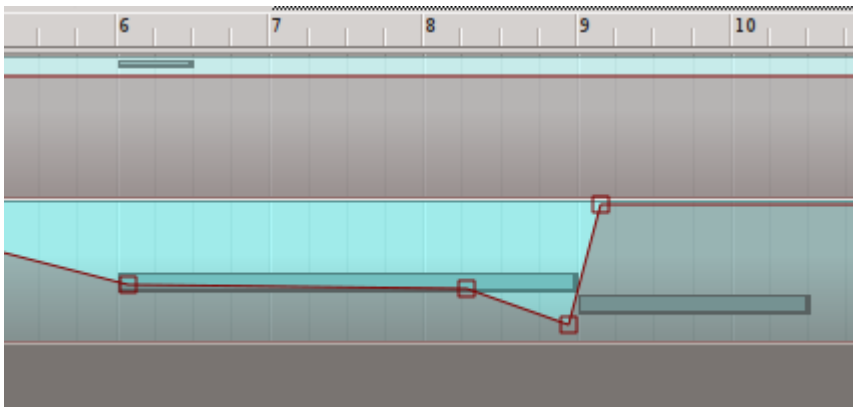
- In the track you wish to automate, click the *Automation* button in the track list. From the pop-up menu, select the attribute of that track that you wish to automate first.
- From the pop-up menu, select the attribute of that track that you wish to automate first. An overlay is placed on the track.



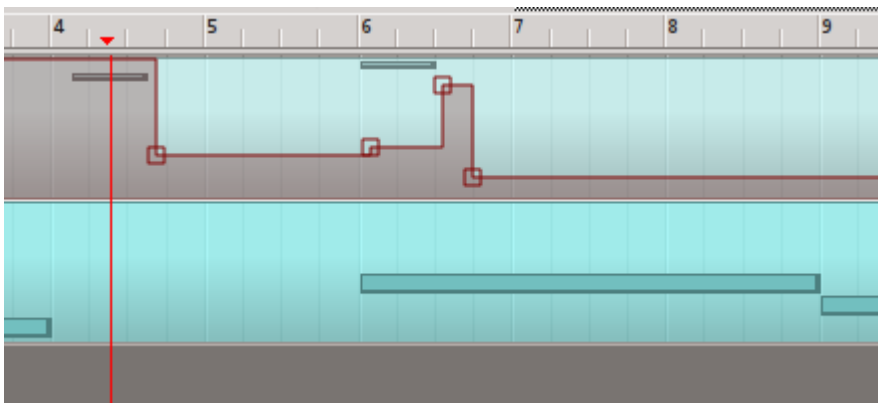
- Click the *Edit* menu to go into the *Select Mode* category, and choose the *Automation* selection tool. Alternately, click the *Automation Selection Tool* icon from the top menu bar.
- Armed with your automation selection tool, click on the automation overlay on your track to create a node. Adjust the volume of this node to be the initial volume of your track. You can either drag the node, or double click it and enter a precise value.
- Click again on the automation overlay to create a second node, and move it to another volume level. For the space between the two nodes, the volume will climb from the starting node level to the ending node level.



If you need a less gradual change, move the nodes closer together, or use more nodes to shape different movements for the mixer.



The automation is customizable via the *Automation* button in the track list, in the *Mode* category, *Hold* places the automation nodes into absolute values until otherwise changed.



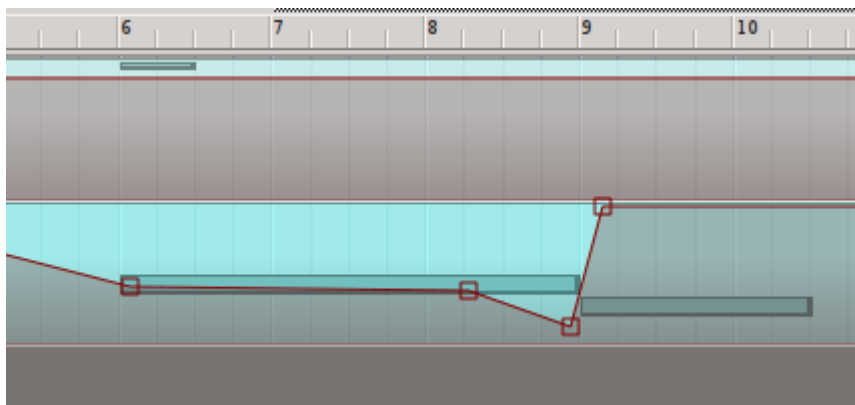
- Linear - represents the automation as even linear progression from one node to the next. This is the default automation mode.
- Spline - same as linear, except with b-splines instead of completely linear progress between nodes.

Automation node values are stored as plain MIDI controller values, which means a default 7-bit resolution; on save/load, the values are quantized to this resolution. Because of this, you may find that the values you originally entered change very slightly upon reloading a session. Despite this, Qtractor should still handle the automation correctly. If you want a higher resolution, you can enable *High resolution plugin automation* from *View > Options... > Plugins > Experimental*, which will give a 14-bit resolution.

Recording Automation



- Logarithmic - same as spline except that the degree of the bezier curves change proportionately to the difference between nodes.



- **Color** - when you automate volume, panning, and an effect all on one track, it can get confusing. Use *Color* to change the shade of the automation overlay so that you can distinguish one set of controls from another. *Lock* prevents accidental changes to automation, especially useful when working on a different automation function on the same track.

Override Automation

If you are a skilled live mixer and want to record your performance in realtime, you can override existing automation decisions, or record new ones, in two different styles, each available in the Automation menu:

- *Play* - with this enabled, you may make realtime changes to existing automation, and upon release of the slider or knob that you are automating, the value returns to the pre-existing value. This may be combined with the *Record* option.
- *Record* - with this enabled, you may make realtime decisions for automation and every move will be recorded, in realtime, as a node in your track. They can be manually changed later, if needed.

If you have the *Play* option enabled, when you release the slider or knob that you are automating, the values will return to their existing level...

Without *Play* enabled, any change you make will persist until you change it again.

Delivery

At the end of the song-writing process, there is the final export. This of course combines all of the sounds being generated from all of the different sources you may be using, filtered through all of the effects and processors you have in place, and places them all into a self-contained file that your audience can play in their media player of choice.

Since everything in Qtractor is routed through its mixer, to combine the sounds into one audio file is as simple as creating a new track and using it to record the *Master Out*:

- Go to the *View* menu and select *Options* to open Qtractor's preferences.
- In the *Options* window, click on the *Audio* tab. Select the kind of audio you'd like to use for your recording. It is generally preferred to export to a lossless format and use that file as the "gold master" from which compressed (ogg vorbis, mp4, mp3, and so on) versions can be produced. A high quality, lossless, free codec is FLAC.

- Click the *OK* button to accept your changes.
- Create a new track in your Qtractor project by clicking the *Track* menu and selecting *Add Track*.
- In the *Mixer* window, right click the plugin window and click the *Inserts* category, choosing *Add Insert*. In the *Insert* window, click the *Return* button. This opens a *Connections* window.
- Click the first (left) *Master In jack* in the left column to the first insert of your new track in the right column, and click the *Connect* button. Repeat this for the second (right) jacks. Close the window when finished.
- You have now routed all of your mixer's outbound sound to the input of your new track. Arm your new track for recording by clicking the *R* button either in the track list or the mixer window.
- Position the transport (playhead) at the beginning of the project (or the beginning of the portion of your song you intend to export).

You may wish to set an in and out point so that the recording stops automatically, or you may prefer to stop the recording manually.

- Press the *Record* button in the top menu bar.

Before recording, double check your connections!

If you have left a microphone plugged in and actively sending input to your *Master in*, for example, you might find later that you've recorded a layer of room tone behind all of your tracks!

- Press the *Play* button to begin playback and recording.
- When recording is finished, select the track by clicking on the new audio clip in the Qtractor project. Click the *Clip* menu and select *Export* to send the clip out as an independent file. Your project is now complete. Create a new project file, and start over with your next song, and enjoy your time with Qtractor.

Appendix A. GNU Free Documentation

License

Version 1.3, 3 November 2008 Copyright © 2000, 2001, 2002, 2007, 2008 Free Software Foundation, Inc.

Everyone is permitted to copy and distribute verbatim copies of this license document, but changing it is not allowed.

0. PREAMBLE

The purpose of this License is to make a manual, textbook, or other functional and useful document “free” in the sense of freedom: to assure everyone the effective freedom to copy and redistribute it, with or without modifying it, either commercially or noncommercially. Secondly, this License preserves for the author and publisher a way to get credit for their work, while not being considered responsible for modifications made by others.

This License is a kind of “copyleft”, which means that derivative works of the document must themselves be free in the same sense. It complements the GNU General Public License, which is a copyleft license designed for free software.

We have designed this License in order to use it for manuals for free software, because free software needs free documentation: a free program should come with manuals providing the same freedoms that the software does. But this License is not limited to software manuals; it can be used for any textual work, regardless of subject matter or whether it is published as a printed book. We recommend this License principally for works whose purpose is instruction or reference.

1. APPLICABILITY AND DEFINITIONS

This License applies to any manual or other work, in any medium, that contains a notice placed by the copyright holder saying it can be distributed under the terms of this License. Such a notice grants a world-wide, royalty-free license, unlimited in duration, to use that work under the conditions stated herein. The “Document”, below, refers to any such manual or work. Any member of the public is a licensee, and is addressed as “you”. You accept the license if you copy, modify or distribute the work in a way requiring permission under copyright law.

A “Modified Version” of the Document means any work containing the Document or a portion of it, either copied verbatim, or with modifications and/or translated into another language. A “Secondary Section” is a named appendix or a front-matter section of the Document that deals exclusively with the relationship of the publishers or authors of the Document to the Document’s overall subject (or to related matters) and contains nothing that could fall directly within that overall subject. (Thus, if the Document is in part a textbook of mathematics, a Secondary Section may not explain any mathematics.) The relationship could be a matter of historical connection with the subject or with related matters, or of legal, commercial, philosophical, ethical or political position regarding them.

The “Invariant Sections” are certain Secondary Sections whose titles are designated, as being those of Invariant Sections, in the notice that says that the Document is released under this License.

If a section does not fit the above definition of Secondary then it is not allowed to be designated as Invariant. The Document may contain zero Invariant Sections.

If the Document does not identify any Invariant Sections then there are none.

The “Cover Texts” are certain short passages of text that are listed, as Front-Cover Texts or BackCover Texts, in the notice that says that the Document is released under this License. A Front-Cover Text may be at most 5 words, and a Back-Cover Text may be at most 25 words.

A “Transparent” copy of the Document means a machine-readable copy, represented in a format whose specification is available to the general public, that is suitable for revising the document straightforwardly with generic text editors or (for images composed of pixels) generic paint programs or (for drawings) some widely available drawing editor, and that is suitable for input to text formatters or for automatic translation to a variety of formats suitable for input to text formatters.

A copy made in an otherwise Transparent file format whose markup, or absence of markup, has been arranged to thwart or discourage subsequent modification by readers is not Transparent.

An image format is not Transparent if used for any substantial amount of text. A copy that is not “Transparent” is called “Opaque”. Examples of suitable formats for Transparent copies include

plain ASCII without markup, Texinfo input format, LaTeX input format, SGML or XML using a publicly available DTD, and standard-conforming simple HTML, PostScript or PDF designed for human modification.

Examples of transparent image formats include PNG, XCF and JPG. Opaque formats include proprietary formats that can be read and edited only by proprietary word processors, SGML or XML for which the DTD and/or processing tools are not generally available, and the machine-generated HTML, PostScript or PDF produced by some word processors for output purposes only.

The “Title Page” means, for a printed book, the title page itself, plus such following pages as are needed to hold, legibly, the material this License requires to appear in the title page. For works in formats which do not have any title page as such, “Title Page” means the text near the most prominent appearance of the work’s title, preceding the beginning of the body of the text. The “publisher” means any person or entity that distributes copies of the Document to the public.

A section “Entitled XYZ” means a named subunit of the Document whose title either is precisely XYZ or contains XYZ in parentheses following text that translates XYZ in another language. (Here XYZ stands for a specific section name mentioned below, such as “Acknowledgements”, “Dedications”, “Endorsements”, or “History”.) To “Preserve the Title” of such a section when you modify the Document means that it remains a section “Entitled XYZ” according to this definition.

The Document may include Warranty Disclaimers next to the notice which states that this License applies to the Document. These Warranty Disclaimers are considered to be included by reference in this License, but only as regards disclaiming warranties: any other implication that these Warranty Disclaimers may have is void and has no effect on the meaning of this License.

2. VERBATIM COPYING

You may copy and distribute the Document in any medium, either commercially or noncommercially, provided that this License, the copyright notices, and the license notice saying this License applies to the Document are reproduced in all copies, and that you add no other conditions whatsoever to those of this License.

You may not use technical measures to obstruct or control the reading or further copying of the copies you make or distribute. However, you may accept compensation in exchange for copies. If you distribute a large enough number of copies you must also follow the conditions in section 3. You may also lend copies, under the same conditions stated above, and you may publicly display copies.

3. COPYING IN QUANTITY

If you publish printed copies (or copies in media that commonly have printed covers) of the Document, numbering more than 100, and the Document’s license notice requires Cover Texts, you must enclose the copies in covers that carry, clearly and legibly, all these Cover Texts: Front-Cover Texts on the front cover, and Back-Cover Texts on the back cover.

Both covers must also clearly and legibly identify you as the publisher of these copies. The front cover must present the full title with all words of the title equally prominent and visible. You may add other material on the covers in addition.

Copying with changes limited to the covers, as long as they preserve the title of the Document and satisfy these conditions, can be treated as verbatim copying in other respects.

If the required texts for either cover are too voluminous to fit legibly, you should put the first ones listed (as many as fit reasonably) on the actual cover, and continue the rest onto adjacent pages.

If you publish or distribute Opaque copies of the Document numbering more than 100, you must either include a machine-readable Transparent copy along with each Opaque copy, or state in or with each Opaque copy a computer-network location from which the general network-using public has access to download using public-standard network protocols a complete Transparent copy of the Document, free of added material.

If you use the latter option, you must take reasonably prudent steps, when you begin distribution of Opaque copies in quantity, to ensure that this Transparent copy will remain thus accessible at the stated location until at least one year after the last time you distribute an Opaque copy (directly or through your agents or retailers) of that edition to the public.

It is requested, but not required, that you contact the authors of the Document well before redistributing any large number of copies, to give them a chance to provide you with an updated version of the Document.

MODIFICATIONS

You may copy and distribute a Modified Version of the Document under the conditions of sections 2 and 3 above, provided that you release the Modified Version under precisely this License, with the Modified Version filling the role of the Document, thus licensing distribution and modification of the Modified Version to whoever possesses a copy of it.

In addition, you must do these things in the Modified Version:

1. Use in the Title Page (and on the covers, if any) a title distinct from that of the Document, and from those of previous versions (which should, if there were any, be listed in the History section of the Document). You may use the same title as a previous version if the original publisher of that version gives permission.
2. List on the Title Page, as authors, one or more persons or entities responsible for authorship of the modifications in the Modified Version, together with at least five of the principal authors of the Document (all of its principal authors, if it has fewer than five), unless they release you from this requirement.
3. State on the Title page the name of the publisher of the Modified Version, as the publisher.
4. Preserve all the copyright notices of the Document.
5. Add an appropriate copyright notice for your modifications adjacent to the other copyright notices.
6. Include, immediately after the copyright notices, a license notice giving the public permission to use the Modified Version under the terms of this License, in the form shown in the Addendum below.
7. Preserve in that license notice the full lists of Invariant Sections and required Cover Texts given in the Document's license notice.
8. Include an unaltered copy of this License.

9. Preserve the section Entitled “History”, Preserve its Title, and add to it an item stating at least the title, year, new authors, and publisher of the Modified Version as given on the Title Page. If there is no section Entitled “History” in the Document, create one stating the title, year, authors, and publisher of the Document as given on its Title Page, then add an item describing the Modified Version as stated in the previous sentence.
10. Preserve the network location, if any, given in the Document for public access to a Transparent copy of the Document, and likewise the network locations given in the Document for previous versions it was based on. These may be placed in the “History” section. You may omit a network location for a work that was published at least four years before the Document itself, or if the original publisher of the version it refers to gives permission.
11. For any section Entitled “Acknowledgements” or “Dedications”, Preserve the Title of the section, and preserve in the section all the substance and tone of each of the contributor acknowledgements and/or dedications given therein.
12. Preserve all the Invariant Sections of the Document, unaltered in their text and in their titles. Section numbers or the equivalent are not considered part of the section titles.
13. Delete any section Entitled “Endorsements”. Such a section may not be included in the Modified Version.
14. Do not retitle any existing section to be Entitled “Endorsements” or to conflict in title with any Invariant Section.
15. Preserve any Warranty Disclaimers.

If the Modified Version includes new front-matter sections or appendices that qualify as Secondary Sections and contain no material copied from the Document, you may at your option designate some or all of these sections as invariant. To do this, add their titles to the list of Invariant Sections in the Modified Version’s license notice.

These titles must be distinct from any other section titles. You may add a section Entitled “Endorsements”, provided it contains nothing but endorsements of your Modified Version by various parties — for example, statements of peer review or that the text has been approved by an organization as the authoritative definition of a standard.

You may add a passage of up to five words as a Front-Cover Text, and a passage of up to 25 words as a Back-Cover Text, to the end of the list of Cover Texts in the Modified Version.

Only one passage of Front-Cover Text and one of Back-Cover Text may be added by (or through arrangements made by) any one entity.

If the Document already includes a cover text for the same cover, previously added by you or by arrangement made by the same entity you are acting on behalf of, you may not add another; but you may replace the old one, on explicit permission from the previous publisher that added the old one.

The author(s) and publisher(s) of the Document do not by this License give permission to use their names for publicity for or to assert or imply endorsement of any Modified Version.

5. COMBINING DOCUMENTS

You may combine the Document with other documents released under this License, under the terms defined in section 4 above for modified versions, provided that you include in the combination all of the Invariant Sections of all of the original documents, unmodified, and list them all as Invariant Sections of your combined work in its license notice, and that you preserve all their Warranty Disclaimers. 88

The combined work need only contain one copy of this License, and multiple identical Invariant Sections may be replaced with a single copy. If there are multiple Invariant Sections with the same name but different contents, make the title of each such section unique by adding at the end of it, in parentheses, the name of the original author or publisher of that section if known, or else a unique number.

Make the same adjustment to the section titles in the list of Invariant Sections in the license notice of the combined work.

In the combination, you must combine any sections Entitled “History” in the various original documents, forming one section Entitled “History”; likewise combine any sections Entitled “Acknowledgements”, and any sections Entitled “Dedications”.

You must delete all sections Entitled “Endorsements”.

6. COLLECTIONS OF DOCUMENTS

You may make a collection consisting of the Document and other documents released under this License, and replace the individual copies of this License in the various documents with a single copy that is included in the collection, provided that you follow the rules of this License for verbatim copying of each of the documents in all other respects.

You may extract a single document from such a collection, and distribute it individually under this License, provided you insert a copy of this License into the extracted document, and follow this License in all other respects regarding verbatim copying of that document.

7. AGGREGATION WITH INDEPENDENT WORKS

A compilation of the Document or its derivatives with other separate and independent documents or works, in or on a volume of a storage or distribution medium, is called an “aggregate” if the copyright resulting from the compilation is not used to limit the legal rights of the compilation’s users beyond what the individual works permit.

When the Document is included in an aggregate, this License does not apply to the other works in the aggregate which are not themselves derivative works of the Document.

If the Cover Text requirement of section 3 is applicable to these copies of the Document, then if the Document is less than one half of the entire aggregate, the Document’s Cover Texts may be placed on covers that bracket the Document within the aggregate, or the electronic equivalent of covers if the Document is in electronic form. Otherwise they must appear on printed covers that bracket the whole aggregate.

8. TRANSLATION

Translation is considered a kind of modification, so you may distribute translations of the Document under the terms of section 4. Replacing Invariant Sections with translations requires

special permission from their copyright holders, but you may include translations of some or all Invariant Sections in addition to the original versions of these Invariant Sections.

You may include a translation of this License, and all the license notices in the Document, and any Warranty Disclaimers, provided that you also include the original English version of this License and the original versions of those notices and disclaimers.

In case of a disagreement between the translation and the original version of this License or a notice or disclaimer, the original version will prevail. If a section in the Document is Entitled “Acknowledgements”, “Dedications”, or “History”, the requirement (section 4) to Preserve its Title (section 1) will typically require changing the actual title.

9. TERMINATION

You may not copy, modify, sublicense, or distribute the Document except as expressly provided under this License. Any attempt otherwise to copy, modify, sublicense, or distribute it is void, and will automatically terminate your rights under this License.

However, if you cease all violation of this License, then your license from a particular copyright holder is reinstated (a) provisionally, unless and until the copyright holder explicitly and finally terminates your license, and (b) permanently, if the copyright holder fails to notify you of the violation by some reasonable means prior to 60 days after the cessation.

Moreover, your license from a particular copyright holder is reinstated permanently if the copyright holder notifies you of the violation by some reasonable means, this is the first time you have received notice of violation of this License (for any work) from that copyright holder, and you cure the violation prior to 30 days after your receipt of the notice.

Termination of your rights under this section does not terminate the licenses of parties who have received copies or rights from you under this License. If your rights have been terminated and not permanently reinstated, receipt of a copy of some or all of the same material does not give you any rights to use it.

10. FUTURE REVISIONS OF THIS LICENSE

The Free Software Foundation may publish new, revised versions of the GNU Free Documentation License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns. See Copyleft.

Each version of the License is given a distinguishing version number. If the Document specifies that a particular numbered version of this License “or any later version” applies to it, you have the option of following the terms and conditions either of that specified version or of any later version that has been published (not as a draft) by the Free Software Foundation.

If the Document does not specify a version number of this License, you may choose any version ever published (not as a draft) by the Free Software Foundation. If the Document specifies that a proxy can decide which future versions of this License can be used, that proxy’s public statement of acceptance of a version permanently authorizes you to choose that version for the Document.

11. RELICENSING

“Massive Multiauthor Collaboration Site” (or “MMC Site”) means any World Wide Web server that publishes copyrightable works and also provides prominent facilities for anybody to edit those works. A public wiki that anybody can edit is an example of such a server. A “Massive

Multiauthor Collaboration” (or “MMC”) contained in the site means any set of copyrightable works thus published on the MMC site. “CC-BY-SA” means the Creative Commons Attribution-Share Alike 3.0 license published by Creative Commons Corporation, a not-for-profit corporation with a principal place of business in San Francisco, California, as well as future copyleft versions of that license published by that same organization.

“Incorporate” means to publish or republish a Document, in whole or in part, as part of another Document. An MMC is “eligible for relicensing” if it is licensed under this License, and if all works that were first published under this License somewhere other than this MMC, and subsequently incorporated

in whole or in part into the MMC, (1) had no cover texts or invariant sections, and (2) were thus incorporated prior to November 1, 2008. The operator of an MMC Site may republish an MMC contained in the site under CC-BY-SA on the same site at any time before August 1, 2009, provided the MMC is eligible for relicensing.

ADDENDUM: How to use this License for your documents

To use this License in a document you have written, include a copy of the License in the document and put the following copyright and license notices just after the title page: Copyright © YEAR YOUR NAME Permission is granted to copy, distribute and/or modify this document under the terms of the GNU Free Documentation License, Version 1.3 or any later version published by the Free Software Foundation; with no Invariant Sections, no Front-Cover Texts, and no Back-Cover Texts. A copy of the license is included in the section entitled “GNU Free Documentation License”.

If you have Invariant Sections, Front-Cover Texts and Back-Cover Texts, replace the “with... Texts.” line with this: with the Invariant Sections being LIST THEIR TITLES, with the Front-Cover Texts being LIST, and with the Back-Cover Texts being LIST.

If you have Invariant Sections without Cover Texts, or some other combination of the three, merge those two alternatives to suit the situation. If your document contains nontrivial examples of program code, we recommend releasing these examples in parallel under your choice of free software license, such as the GNU General Public License, to permit their use in free software.

Appendix B. Various and Sundry

Some software is not required for Qtractor to run, but are useful to have and can be run along with Qtractor. Optional installs include:

- [QJackCtl](#) - a user-friendly control panel for JACK, which can help you fine tune latency settings, and wire your virtual instruments to your virtual mixer. The JACK subsystem will be installed automatically by your distribution; check to ensure that QJackCtl is installed. Being written by the same programmer, many control panels available in QJackCtl are also available from within QJackCtl, but unless you very familiar with jackd, you will probably find JACK easier to use with QJackCtl
- [Qsynth](#) - a user-friendly controller for the Fluidsynth soft synth and a good example (for its familiarity) of how soft synths can be used external of the Qtractor interface and yet still send sound to Qtractor
- [Fluidsynth-DSSI](#) - the Fluidsynth soft synth in plugin-form so that soundfonts can be used from within the Qtractor interface

- [WhySynth](#) - a DSSI plugin that ships with a number of high-quality pre-sets, providing quick satisfaction as well as plenty of oscillators and dials to allow you to create your own patches
- [Steve Harris LADSPA Plugins](#) - the famous must-have set of audio plugins. The LADSPA versions have been around for years, the newer LV2 versions are also available and [LV2](#) support is starting to become standard, so feel free to try the newer ones instead.
- [CALF audio plugin pack](#) - provides a compressor, chorus, reverb, flanger, delay, and lots more, as either DSSI, LADSPA, or LV2 plugins. They are easy to use and feature attractive interfaces.

Appendix C. General MIDI

Instrumentation

This is the standardized list of General MIDI channels and the corresponding instrumentation. If you are composing using MIDI and want your piece to call a standard set of instruments so that it will sound the same across all systems, **this is the list to use.**

Distributing MIDI files of renditions of popular songs was fairly common in the early days of the Internet, when it was considered technologically advanced to have a MIDI song automatically play in the background when someone visited one's homepage. With General MIDI, it could be ensured that the song sounded more or less the same no matter whose system it was.

Of course, this is rather rarely done now. It is far more common to use MIDI to trigger sounds and instruments in any format that works best for you, then to record the music and re-distribute the ogg or mp3 version.

Even so, this is a good reference list since General MIDI was widely used at one point.

1. Acoustic Grand Piano
2. Bright Acoustic Piano
3. Electric Grand Piano
4. Honky-tonk Piano
5. Electric Piano 1
6. Electric Piano 2
7. Harpsichord
8. Clavinet
9. Celesta
10. Glockenspiel
11. Music Box
12. Vibraphone

13. Marimba
14. Xylophone
15. Tubular Bells
16. Dulcimer
17. Drawbar Organ
18. Percussive Organ
19. Rock Organ
20. Church Organ
21. Reed Organ
22. Accordion
23. Harmonica
24. Tango Accordion
25. Acoustic Guitar (nylon)
26. Acoustic Guitar (steel)
27. Electric Guitar (jazz)
28. Electric Guitar (clean)
29. Electric Guitar (muted)
30. Overdriven Guitar
31. Distortion Guitar
32. Guitar harmonics
33. Acoustic Bass
34. Electric Bass (finger)
35. Electric Bass (pick)
36. Fretless Bass
37. Slap Bass 1
38. Slap Bass 2
39. Synth Bass 1
40. Synth Bass 2
41. Violin

42. Viola
43. Cello
44. Contrabass
45. Tremolo Strings
46. Pizzicato Strings
47. Orchestral Harp
48. Timpani
49. String Ensemble 1
50. String Ensemble 2
51. Synth Strings 1
52. Synth Strings 2
53. Choir Aahs
54. Voice Oohs
55. Synth Voice
56. Orchestra Hit
57. Trumpet
58. Trombone
59. Tuba
60. Muted Trumpet
61. French Horn
62. Brass Section
63. Synth Brass 1
64. Synth Brass 2
65. Soprano Sax
66. Alto Sax
67. Tenor Sax
68. Baritone Sax
69. Oboe
70. English Horn

- 71. Bassoon
- 72. Clarinet
- 73. Piccolo
- 74. Flute
- 75. Recorder
- 76. Pan Flute
- 77. Blown Bottle
- 78. Shakuhachi
- 79. Whistle
- 80. Ocarina
- 81. Lead 1 (square)
- 82. Lead 2 (sawtooth)
- 83. Lead 3 (calliope)
- 84. Lead 4 (chiff)
- 85. Lead 5 (charang)
- 86. Lead 6 (vox)
- 87. Lead 7 (fifths)
- 88. Lead 8 (lead bass)
- 89. Pad 1 (new age)
- 90. Pad 2 (warm)
- 91. Pad 3 (polysynth)
- 92. Pad 4 (choir)
- 93. Pad 5 (bowed)
- 94. Pad 6 (metal)
- 95. Pad 7 (halo)
- 96. Pad 8 (sweep)
- 97. FX 1 (rain)
- 98. FX 2 (soundtrack)
- 99. FX 3 (crystal)

100.FX 4 (atmosphere)
101.FX 5 (brightness)
102.FX 6 (goblins)
103.FX 7 (echoes)
104.FX 8 (sci fi)
105.Sitar
106.Banjo
107.Shamisen
108.Koto
109.Kalimba
110.Bag pipe
111.Fiddle
112.Shanai
113.Tinkle Bell
114.Agogo
115.Steel Drums
116.Woodblock
117.Taiko Drum
118.Melodic Tom
119.Synth Drum
120.Reverse Cymbal
121.Guitar Fret Noise
122.Breath Noise
123.Seashore
124.Bird Tweet
125.Telephone Ring
126.Helicopter
127.Applause
128.Gunshot

Index

A

- Amsynth
- Automation

B

- Bouncing

C

- Calf
- Clips
- Compressors

E

- Editing
- Editor
- Equalizers
- Effects
- Exporting

F

- Fades

H

- Hydrogen

I

- Importing audio
- Importing data
- Installation from source

J

- Jamin

L

- Limiter
- Loops

M

- Merging
- Modes
- Moving
- MIDI
- Mixdown
- Mixer

N

- Noise removal
- Normalization

O

- Overview

P

- Preferences
- Punch-in
- Playback
- Plugins

Q

- Quick start
- Qsynth

R

- Reverb
- Range markers
- Range, rectangle

- Realtime kernel
- Recording
- Recording a live mix

S

- Synths
- set_rlimits
- Splitting
- Stereo
- Selection tool
- Sends / returns
- Sequencers
- System requirements

T

- Track order
- Troubleshooting
- Tools

U

- USB controller

V

- VST sdk

W

- Waveforms

Z

- Zoom

Resources

- Qtractor book at slackermidia: <http://slackermidia.info/html/qtractor-book.html>