MIDI Tools

Version 2.1.0

MANUAL

MOUNTAIN UTILITIES

https://mountainutilities.eu/

CONTENTS

1. Overview

MIDI Tools is a utility by Mountain Utilities which (as its name implies) provides various tools for working with MIDI. It is available for macOS and Windows. It is free, although donations are more than welcome.

MIDI Tools provides the following facilities:

- Real-time MIDI input and output monitors.
- Capturing, filtering and saving MIDI input messages.
- On Windows: the creation of MIDI thru connections (e.g. from/to virtual MIDI I/O ports).
- A virtual MIDI controller board, capable of receiving and sending control change messages.
- A virtual MIDI keyboard (outputting note messages), controlled by the computer mouse. This includes a primitive arpeggiator.
- A window from which you can send NRPN and RPN messages.
- A window from which you can send MIDI Program Change messages, optionally prefixed by Bank Select MSB/LSB messages.
- A window from which you can send MIDI System Exclusive, System Common and System Real-time messages for testing purposes.
- A powerful editor/uploader/downloader for MIDI System Exclusive messages.
- You can extract any System Exclusive messages from a standard MIDI (.mid) file and export them to a System Exclusive (.syx) file.
- A MIDI data byte converter/calculator.
- Virtual displays for Mackie Controls (much like those provided by the Huskervu utility).

2. Version history

Version 2.1.0 (Windows 2018-06-14; macOS 2019-01-10)

All editions:

- The MIDI Tools update mechanism allows you to specify in which update types you are interested: alpha (development) versions, ('public') beta versions, release candidates and release versions.
- Since the Mountain Utilities website has changed from the HTTP protocol to HTTPS, all links to the Mountain Utilities website have changed to HTTPS too.
- The Mackie monitors now accept display messages from *all* Mackie models (such as the Mackie Control XT).

macOS edition:

- The application no longer refuses to run if a MIDI input or output device contains a 'weird' character in its name, manufacturer or model.
- In the MIDI data calculator, the position of a 'Binary' label was corrected.

Windows edition:

- The exe file of the installer is no longer offered within a zip file, but directly.
- The installer automatically installs the edition of the application that matches the operating system: the 32-bit edition on 32-bit Windows, the 64-bit edition on 64-bit Windows. (Previously there was only a 32-bit edition, which was installed on 32- and 64-bit Windows alike.)
- Portable 32- and 64-bit editions are available. The 32-bit edition runs on 32- and 64-bit Windows, the 64-bit edition only on 64-bit Windows.

Changes to this manual:

- In accordance with Apple renaming OS X to macOS, all occurrences of 'OS X' have been replaced with 'macOS'.
- CopperLan has been added to the list of MIDI pipes in §5. (Thanks to JAPP for bringing CopperLan to my attention.)

Version 2.0.1.1 (2016-02-26; macOS edition only)

Bug fix: Opening the 'Roland System Exclusive message(s)' dialog box (from the 'System Exclusive messages' window) no longer removes the 'MIDI Tools' pull-down menu from the menu bar.

Version 2.0.1 (2016-02-08)

This version corrects a few minor problems that were introduced in version 2.0.0:

- The 'Restart with default setup' operation automatically enables certain MIDI devices again.
- *macOS edition only:* Updates to the timer window are taken seriously again.
- *Windows edition only:* When the application is started for the very first time or after 'Restart with default setup', the main window is positioned at the intended fixed position again (rather than at a more or less random position).

Version 2.0.0 (2016-02-01)

New features:

- *Windows edition only:* If a previous version of the application has been installed, the installer skips the dialog box in which you can set the destination folder and the dialog box in which you can set the program group in the Windows menu. So if you wish to install to a different folder or program group, you must uninstall the previous version first.
- Maintenance of multiple application setups (such as window positions and the enabled/disabled states of MIDI devices) via setup ('.stp') files. See the Setup submenu: on macOS this is in the MIDI Tools pull-down menu, on Windows in the File pull-down menu of the main window.

Note that opening a previously saved setup file involves restarting the application. You can also start the application with a setup file as a command line parameter: -s *setupfile* on macOS, /s *setupfile* on Windows.

- Maintenance of the application's window positions via desktop ('.dsk') files; see View ⇨ Desktop in the main window. Unlike opening a setup file, opening a desktop file does *not* involve restarting the application.
- You can sanitize the lists of most-recently opened/saved files in the menus via two operations: 'Remove absent files from list' and 'Clear list'.
- MIDI controllers window: the control bars now respond to Control Change messages received at the selected channel of the selected input device.
- MIDI controllers window: in the Options dialog box a button selecting controllers 0-31 has been added.
- MIDI System messages window: System Exclusive messages can be sent repeatedly (via Start and Stop). This can be useful for stress-testing the SysEx buffer implementations of MIDI devices and applications.

New features in the MIDI System Exclusive window (many thanks to Royce Craven for his input):

- The window now has three panels: MIDI, Probe and Toolbar, each of which can be hidden via the View pull-down menu.
- On the Probe panel, an editbox (with a history list) allows you to type one or more SysEx messages. These messages can be sent to a MIDI device, and the history list can be saved to and loaded from a text ('.txt') file.
- You can prevent the grid from automatically scrolling to the latest received message by clearing the new View ⇨ 'Scroll to new message' setting.
- The vertical order of the representations of the message bytes in the grid has changed. From top to bottom the order is now: decimal, hexadecimal, binary, character. (This order is the same as in the MIDI data calculator window, accessible from the main window's View pull-down menu.)
- The old Edit operation (allowing you to edit the selected messages as text lines) is still available, but has been renamed to 'Edit message(s)' and its icon has changed to a 'T'.
- The 'E' icon is now associated with the new 'Edit byte(s)' operation. This operation allows you to edit the selected byte(s) in the message grid individually via a dialog box, offering three editing formats: decimal, hexadecimal and binary. The corresponding ASCII character is also displayed.
- Quite a few keyboard shortcuts have changed.
- 'Correct Roland checksum(s)' can be applied on demand to the message(s) selected in the probe edit box or the grid.
- If 'Auto-correct Roland checksum(s)' in the Edit pull-down menu has been ticked, the 'Edit byte(s)' and 'Edit message(s)' operations automatically correct any invalid Roland checksums resulting from your edits to the grid. Similarly, if 'Auto-correct Roland checksum(s)' in the Probe pull-down menu has been ticked, the message(s) in the probe editbox is/are automatically corrected when you execute Send.

Bug fixes (macOS and Windows editions):

- Mackie monitors: the height of the window now updates immediately after its font size has changed.
- MIDI System messages window: the Start button for Timing Clock now uses the correct interval.

Bug fixes (macOS edition only):

- Incoming MIDI System Exclusive messages are now processed correctly. (Previously the terminating F7 could get cut off, leading to error message 'Not enough MIDI SysEx input buffers'.) *Many thanks to adriaanhendrik for reporting this bug and assistance in fixing it.*
- Non-SysEx messages coming in in quick succession are now processed correctly. (Previously such messages would get skipped and lead to error message 'Non-SysEx MIDI input message too long'.)

- MIDI input messages window: the actual maximum number of messages in the grid now keeps matching the value stipulated in the Options dialog box.
- MIDI System Exclusive window: the Edit ⇨ Insert operation is now executed upon Ctrl+N. (Previously it was associated with Ctrl+I, but this was (and is) already associated with File ⇨ Insert.)
- The menu bar now consistently shows the active window's pull-down menus. So if the active window doesn't *have* any pull-down menus, nothing is shown; previously in this situation, the pull-down menus of the previously active window would still be shown.

Version 1.8.0 (2015-09-23)

- New: a window from which you can send NRPN and RPN messages. Useful for testing purposes. See §10 for more information.
- In the 'Window list' dialog box you can make all windows visible in one operation.

Version 1.7.0 (2015-03-05)

- First version also available for macOS.
- Many internal improvements to MIDI input and output communication. In particular, the input system for SysEx messages has been redesigned completely, one consequence being that the maximum SysEx message length is now fixed at 65536 bytes, hence it is no longer editable via the Input tab of the MIDI devices dialog box.
- On the MIDI keyboard, Middle C has a blue border.
- The MIDI input messages window can display message times in various formats: ((((days:)hours:)min:)sec.)ms.
- Several new features in the MIDI System Exclusive window:
  1. You can export the selected messsage(s) to a text file, optionally including the message indexes and lengths.
  2. The message bytes are displayed in a grid, in which you can highlight specific columns by right-clicking. Moreover, the bytes can be displayed in various formats simultaneously: binary, hexadecimal (the default), decimal and ASCII character.

  See §12 for more information.
- MIDI Tools no longer refuses to start if system.ini doesn't exist in the Windows system folder.
- The Donate item in the main window's Help pull-down menu no longer opens a dialog box, but makes your web browser open the Donate page at the Mountain Utilities web site.

Version 1.6.0 (2014-02-01)

- A window in which you can perform MIDI data byte conversions.
- MIDI Tools uses the new Mountain Utilities web site at mountainutilities.eu in links and its update mechanism.
- In this manual:
  - Many screenshots of windows and dialog boxes.
  - Several new sections dedicated to particular windows.
  - New information about MIDI Yoke's behavior under Windows versions with UAC.

Version 1.5.0 (2012-12-18)

- A significant overhaul of the MIDI controllers window, with many new display options.
- The MIDI keyboard window can now switch a key off by a Note On message with Velocity 0 (as before) or by a Note Off message (with customizable 'offset velocity'). Press the 'Velocity options' button to edit all velocity-related settings; all these settings are retained between runs of MIDI Tools.
- From the MIDI System messages window you can now send an 'Identity Request' message (a specific System Exclusive message), to which certain MIDI devices reply with an 'Identity

Reply' message.

- In the MIDI System messages window, any button's keyboard shortcut (cf. the underlined character) now selects the button but does *not* send the MIDI message associated with the button. (In version 1.4.0, some button hotkeys did *not* select the button but *did* send the MIDI message.)
- A small bug fix in the 'MIDI System Exclusive message(s)' dialog box (opened from the MIDI System Exclusive messages window): if the sequence contains an invalid byte, the dialog box popping up when you press OK now shows the correct SysEx message number.

Version 1.4.0 (2012-11-30)

- New: the 'MIDI System messages' window, which allows you to send System Exclusive, System Common and System Real-time messages for testing purposes.
- In the MIDI keyboard window, a bug (introduced in version 1.3.0) concerning arpeggios and chords was fixed.

Version 1.3.0 (2012-06-11)

New features:

- The 'MIDI program changer' window: this allows you to send MIDI Program Change messages prefixed by Bank Select MSB/LSB messages, in any combination.
- In the MIDI keyboard window, each note's loudness ('onset velocity') is determined by the mouse's vertical position on the key.
- LoopMIDI has been added to the list of known virtual MIDI ports ('pipes').

  As in previous versions, when you start MIDI Tools for the first time you must choose to disable or enable all MIDI pipes in MIDI Tools, and this now includes all loopMIDI ports, or rather those ports that have 'loopMIDI' in their names — as suggested by loopMIDI. However, loopMIDI allows you to change these names completely — if you do so, MIDI Tools won't recognize them as MIDI pipes, so then you can only enable them manually. i.e. via the MIDI devices dialog box.

Improvement:

- Windows that were open when the previous session of MIDI Tools terminated reappear exactly where they were. (In previous versions of MIDI Tools, even a window on a secondary monitor always reappeared on the primary display, even when the secondary monitor was still available.)

  This new behavior has one potentially problematic consequence: when you remove a monitor or reduce the screen resolution, windows may become invisible upon a restart of MIDI Tools. To remedy this, you can use the new 'Make fully visible' operation in the 'Window list' dialog box (accessible via the main window's View pull-down menu or the Alt+0 key combination).

Version 1.2.1 (2011-03-10)

The Mackie monitors no longer display garbage.

Version 1.2.0 (2011-01-09)

- Unicode support: in most text edit boxes in the program (e.g. for file names) you can now enter any 'international' characters. (Technical note: MIDI Tools now saves its ini file in UTF-8 format, starting with the three-byte UTF-8 BOM (byte order mark); however, it can still read the plain ASCII/ANSI ini files of previous versions of MIDI Tools.)
- As a consequence of its new Unicode support, MIDI Tools no longer runs under Windows 95, 98 or Me, because these operating systems do not support Unicode. (If you require a version of MIDI Tools that runs under these operating systems, please send a message to the contact address at the Mountain Utilities web site.)
- The new Restart operation terminates MIDI Tools and automatically starts a new instance of it.

There is also a version of Restart that restarts MIDI Tools with its default setup.

Version 1.1.1 (2010-10-30)

- Several improvements to the Mackie monitors: they now stay on top of the windows in other applications consistently and they have a few new settings (which can be customized via their dialog boxes).
- You can keep MIDI Tools' main window on top of MIDI Tools' other windows and other applications.

Version 1.1.0 (2010-09-26)

- Up to four virtual displays showing the text messages which certain MIDI applications send to (pseudo-)Mackie Controls. This is useful if a pseudo-Mackie Control doesn't have a display itself.
- The link to the Mountain Utilities web site installed in the Windows start menu has been updated.

Version 1.0.3 (2009-11-02)

All references to the Mountain Utilities web site have been updated from 'home.hetnet.nl' to 'home.kpn.nl'. Hence the automatic update mechanism will work again.

Version 1.0.2 (2009-08-06)

- The package uses a different installer (Inno Setup instead of InstallShield). Consequently, the zip file's size has gone down from 4 MiB to 1 MiB! You may also note a few minor differences in the installation procedure.
- Fixed a bug in the update checking mechanism that could cause MIDI Tools to draw the wrong conclusion as to the availability of a new version.

Version 1.0.1 (2009-07-15)

New features:

- This manual.
- From the System Exclusive window, you can extract the System Exclusive messages from a standard MIDI file and save them to a System Exclusive file.
- Facilities for automatic and manual checking whether an update of MIDI Tools is available from the Mountain Utilities web site.

Version 1.0.0 (2009-04-30)

First published version.

3. Computer requirements

To run MIDI Tools, your computer must comply with the following requirements:

- Processor: Any Intel 80486- or Pentium-compatible CPU. Processor speed is relatively unimportant.

- Operating system:
  - macOS: any version for the Intel x86-architecture.
  - Windows: as of version 1.2.0, MIDI Tools only runs under Windows operating systems that support Unicode, such as Windows 2000, XP, Vista, 7, 8 and 10.

- An SVGA-compatible graphical card and monitor:
  - The screen size should be at least 800 × 600 pixels. (However, some of the bigger windows and dialog boxes may be cut off if the screen is too narrow, so 1024 × 768 is the *practical* minimum.)
  - For best results, the color depth should be at least 16 bits. (At a depth of only 256 colors, some colors aren't rendered as intended.)

- A mouse.

- Free hard disk space: about 12 MiB on macOS and 8 MiB on Windows.

- RAM: when running, MIDI Tools normally occupies roughly 16 MiB on macOS and 5 MiB on Windows.

4. Installation of MIDI Tools

To install MIDI Tools on your computer, proceed as follows:

macOS:
1. Download MidiTls-*a.b.c*.dmg (where *a.b.c* stands for the actual version number) to your computer from the MIDI Tools page at the Mountain Utilities web site (https://mountainutilities.eu/miditools).
2. Open the dmg file in Finder, and drag-and-drop the MIDI Tools icon on the Applications icon. If a previously installed version of MIDI Tools exists in the Applications folder, you are asked what you want to do: it's best to select Replace.
3. Right-click the dmg file's 'disc' icon on the right side of the desktop and run Eject from the local menu.

Windows:
On Windows, three editions are available: an installer, a 32-bit portable edition and a 64-bit portable edition:

Installer:
1. Download miditls-*a.b.c*-install.exe (where *a.b.c* stands for the actual version number) to your computer from the MIDI Tools page at the Mountain Utilities web site (https://mountainutilities.eu/miditools).
2. Run miditls-*a.b.c*-install.exe and follow its instructions. The installer automatically installs the edition of the actual application (MidiTls.exe) that matches the operating system: the 32-bit edition on a 32-bit Windows system, the 64-bit edition on a 64-bit Windows system.

   Note: the installation includes an uninstaller. It can be run from the Windows Start Menu via Programs ➜ Mountain Utilities ➜ MIDI Tools, or via Settings ➜ Control Panel ➜ 'Add or Remove Programs' (Windows XP) or 'Programs and Features' (Windows Vista and later).

   Note that when you install a new version of MIDI Tools, you do *not* have to uninstall any previously installed version first: the old version will be replaced with the new version automatically.

Portable (32- or 64-bit):
1. Download miditls-*a.b.c*-x*nn*-portable.zip (where *a.b.c* stands for the actual version number and x*nn* is 'x64' or 'x86') to your computer from the MIDI Tools page at the Mountain Utilities web site (https://mountainutilities.eu/miditools). The 'x64' (64-bit) edition only runs on 64-bit Windows, the 'x86' (32-bit) edition on 32- and 64-bit Windows.
2. Unzip miditls-*a.b.c*-x*nn*-portable.zip completely (maintaining the zip file's tree structure) to any folder to which MIDI Tools itself (MidiTls.exe) will have write-access (which is necessary for its configuration files). Crucially, in Windows Vista and later you should *not* unzip to a subfolder of C:\Program Files or C:\Program Files (x86), since these folders are subject to Windows' UAC (User Access Control), which means that MIDI Tools would *not* have write-access.

Running MIDI Tools itself

After installation, you can start MIDI Tools itself: MIDI Tools.app on macOS (from the Applications folder), MidiTls.exe on Windows (e.g. via the Windows start menu).

If you have never run MIDI Tools (in any version) from the installed operating system before, the program notifies you that it can't find your configuration. This is normal: the program saves its configuration file (miditls.ini in macOS, MidiTls.ini in Windows) whenever the program *terminates*,

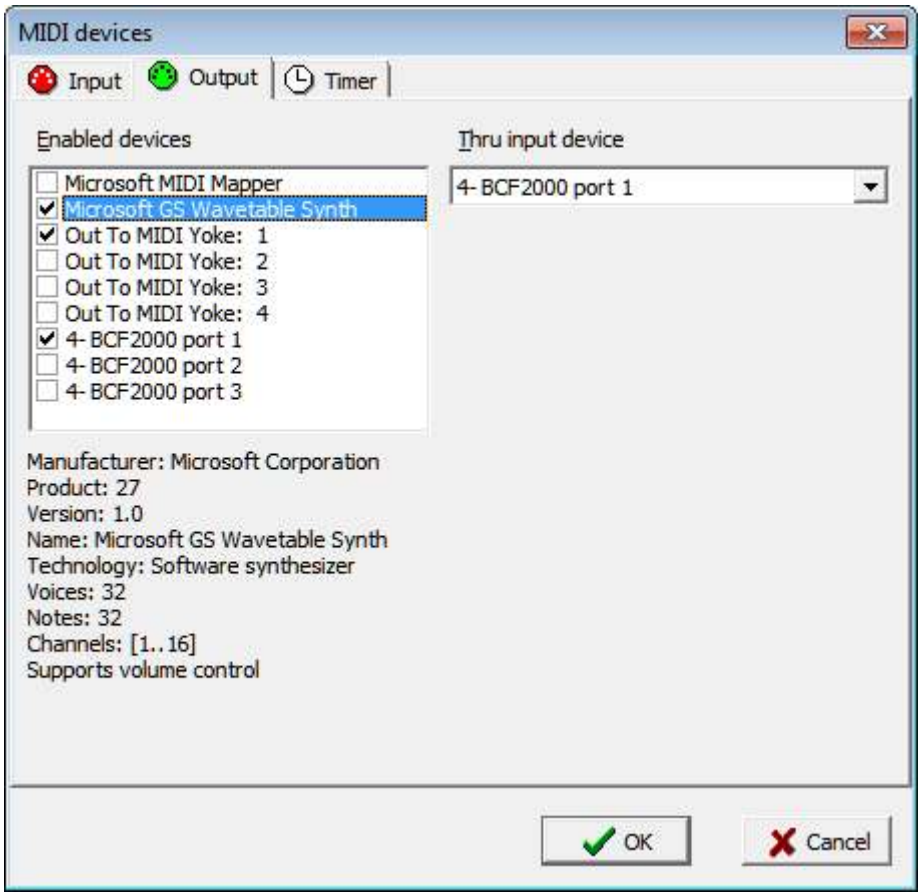
so obviously this configuration file doesn't exist yet when you start the program for the very first time. The program also notifies you if the configuration file does exist but belongs to a previous version; all existing settings are retained.

5. MIDI setup

MIDI Tools can only communicate with MIDI input and output devices that are *enabled*.

When MIDI Tools is started for the first time, it enables all MIDI input and output devices, except that the Windows edition skips the MIDI Mapper and any software synthesizers. However, you are advised to fine-tune this default setup to suit your particular needs. For instance, if you want to run other programs using MIDI devices simultaneously, it may be a good idea to keep as many MIDI devices disabled as you can in MIDI Tools, in order to avoid MIDI device access conflicts.

From the main window's Options pull-down menu, 'MIDI devices' opens a dialog box in which you can select the MIDI devices to which MIDI Tools connects:



Tip (Windows only): In this dialog box you can also set up a 'soft MIDI thru' link, by connecting an enabled MIDI input to an enabled MIDI output device. Any MIDI message received by the MIDI input device is then immediately passed on to the MIDI output device. In fact, you can set up as many links as there are devices; the only restriction is that each device can have only one connection.

MIDI pipes

Of particular interest is the enabling/disabling of 'MIDI pipes'. 'MIDI pipe' is my term for what is commonly known as a 'virtual MIDI device'. This is software that manifests itself as a virtual MIDI output port plus a virtual MIDI input port: the driver passes any MIDI data sent to the output port to

the input port. Hence, when a computer program sends data to the virtual MIDI output port, this data can be picked up at the corresponding input port by any other computer program. Thus, a MIDI pipe allows inter-program MIDI communication. If a MIDI pipe driver is 'multi-client', we can connect more than one program (up to a particular maximum) to the same virtual output or input port.

To my knowledge, the following free MIDI pipe drivers are available (please contact me if you know any others):

- **Hubi's Loopback Device**: 4 multi-client pipes, but for Windows 9*x* only.

- **Sony/Sonic Foundry Virtual MIDI Router**: 4 single-client pipes. Windows 32-bit only.

- **Hurchalla Maple**: 12 single-client pipes. Windows 32-bit only.

- **LoopBe1**: only 1 multi-client pipe, so not very useful. (No, you can't install more than one copy!) Windows 32-bit only.

- **LoopBe30**: 30 multi-client pipes, but the trial version only works for a brief period, and the full version is not free. Windows 32-bit only.

- **MIDI Yoke** (http://www.midiox.com/): the NT (/2000/XP/Vista/7/8(?)) version allows up to 16 multi-client pipes, so understandably this has been the most popular MIDI pipe driver for 32-bit Windows versions.

  Problems:

  1. MIDI Yoke's NT version, even though it is 32-bit, *can* be installed under 64-bit operating systems, but (reportedly) the pipes are only accessible to 32-bit DAWs, not to 64-bit DAWs.
  2. To work with MIDI Yoke correctly under Windows versions with UAC (User Access Control) you must apply a manual tweak:

     The MIDI Yoke installer tries to create MIDI Yoke's configuration file (MYOKENT.INI) in C:\Windows, but the operating system doesn't allow this and actually creates it in C:\Users*Username*\AppData\Local\VirtualStore\Windows. On the other hand, the MIDI Yoke configuration applet under Control Panel *does* have write access to C:\Windows and will create a second copy of MYOKENT.INI there when you change the settings. However, the latter file will never be never seen by the *driver* (because the operating system keeps redirecting it to the copy in ...\VirtualStore\Windows); in other words, the driver 'won't listen to you.'

     To fix this, you must manually remove MYOKENT.INI from C:\Users*Username*\AppData\Local\VirtualStore\Windows or move it to C:\Windows, using administrator rights.
  3. As discussed below, the NT version of MIDI Yoke may slow down the termination of MIDI Tools.

- **CopperLan** (http://www.copperlan.org/):
  Available for macOS and Windows (32- and 64-bit).
  This is primarily a MIDI-over-Ethernet system, so it's a bit of overkill if you only need local MIDI pipes. (CopperLan 1.4 for Windows installs *three* drivers, if I remember correctly!)
  It offers up to 32 virtual MIDI input ports and 32 virtual MIDI output ports; by default none of these are set up as 'pipes', but you can manually connect any output to any input (although the idiosyncratic user interface makes this much more difficult than it should be).

- **loopMIDI** (http://www.tobias-erichsen.de/):
  This allows you to create and destroy any number of MIDI pipes on the fly.
  Simple and effective, so probably the best choice on modern Windows versions (particularly 64-bit versions, given MIDI Yoke's problems on those).

If configured improperly, MIDI pipes can easily cause problematic MIDI signal paths. There are several dangers:

Duplication:
If a there is first a *split* in the signal path, and then a *merge*, two or more copies of the same MIDI message arrive at the end of the signal path (i.e. the target MIDI device). This is time-consuming in all cases, but — even worse — it can mess up communication with certain MIDI devices.

Feedback:
In general, feedback involves the return of a sent MIDI message to the same MIDI hardware device or computer program that sent the message. Obviously this needlessly slows down operation, although it isn't necessarily disastrous. However, there may also be more sinister effects. For instance, feedback may interfere with MIDI Tools' communication with the device.

At the very least you should normally avoid enabling both the output port and the input port of the same MIDI pipe in the same program (e.g. MIDI Tools), because by definition anything you send to a MIDI pipe's *output* port (the pipe's starting point) is returned at the corresponding MIDI pipe's *input* port (the pipe's end point). So for instance, if you enable both 'Out To MIDI Yoke: 1' and 'In From MIDI Yoke: 1' in a program, then any MIDI data the program sends to 'Out To MIDI Yoke: 1' is returned to the program at 'In From MIDI Yoke: 1'. This type of feedback is usually undesired, except perhaps for monitoring purposes.

The most severe type of feedback occurs when the sender/recipient actually *re-sends* the returned MIDI message: this leads to an infinite loop, which may well grind the sender/recipient (and indeed the whole computer) to a virtual standstill.

In MIDI Tools this horror can happen if you enable the MIDI Thru feature in the MIDI devices dialog box for an input-output pair already exhibiting feedback. For instance, if you activate MIDI Thru from 'In From MIDI Yoke: 1' to 'Out To MIDI Yoke: 1' in MIDI Tools' MIDI devices dialog box, then any MIDI data sent to 'Out To MIDI Yoke: 1' not only comes back to MIDI Tools at 'In From MIDI Yoke: 1' (via MIDI Yoke's pipe 1), but is then automatically *re-sent* from 'In From MIDI Yoke: 1' to 'Out To MIDI Yoke: 1' via the MIDI Thru feature, in principle ad infinitum, although MIDI Yoke does perform some checks that spot and kill the infinite loop — but still...

Close delay:
This problem only occurs with MIDI Yoke NT (but *not* with MIDI Yoke *for Windows 95/98/Me*): closing any MIDI Yoke NT 1.75 *input* port causes a delay of 1 second. (Certain earlier versions even 3 seconds.)

Concerning MIDI Tools this is mainly relevant during program exit. In principle MIDI Tools terminates almost instantly upon exit, but when all the input ports of MIDI Yoke NT 1.75 are enabled, termination of MIDI Tools takes some 16 seconds longer than normal! Therefore you should disable as many MIDI Yoke NT input ports as possible in MIDI Tools' MIDI devices dialog box, i.e. any MIDI Yoke NT input ports that MIDI Tools itself doesn't use. (Note that you can still use any MIDI ports disabled in MIDI Tools in *other* programs!)

To help you avoid some of the serious problems discussed above, MIDI Tools takes the following steps:

- On *first* startup, if MIDI Tools detects any of the MIDI pipes listed above (excluding

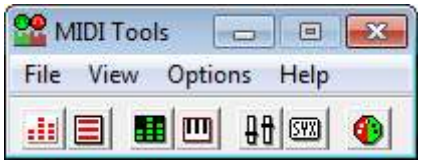
CopperLan, since by default its virtual ports aren't interconnected), it asks you if you want to enable the I/O devices of these pipes. It's best to answer *No* (to avoid feedback loops, and to avoid MIDI Yoke NT's close delays during MIDI Tools' exit procedure), unless some other program (e.g. MIDI-OX) is routing one of your MIDI devices through a MIDI pipe.

Note that CopperLan's virtual MIDI input and output ports are *not* co

- On *every* startup, MIDI Tools optionally warns you if any MIDI Yoke NT input ports are enabled and thereby cause extra delays during termination of MIDI Tools. You can enable/disable this warning on the Input tab of the MIDI devices dialog box.

6. The main window

MIDI Tools' main window only consists of a pull-down menu and a toolbar:



The toolbar merely contains a number of buttons duplicating some of the most useful menu items.

The menu provides the following operations:

MIDI Tools (macOS)/File (Windows) ➜ Restart:
Terminates MIDI Tools and starts a new instance of it. This is particularly useful after the MIDI device configuration has changed while MIDI Tools has been running, either because a MIDI device (often a USB-based one) has become invalid or because a new one has become available: restarting MIDI Tools updates MIDI Tools' device list to the new configuration.

MIDI Tools (macOS)/File (Windows) ➜ Restart with default setup:
Functions like Restart (see above), with two differences:

- Since this operation is somewhat 'momentous', a dialog box requires you to confirm that you indeed want to do this.
- The default setup file MidiTls.stp is deleted before the restart, so that all setup values (such as MIDI I/O device settings and window positions/sizes) are restored to their defaults. Consequently, MIDI Tools' new instance behaves as if you have never run the application before.

This operation may be useful when some setup problem has developed that you find yourself unable to fix quickly otherwise. However, the restoration of the default setup also has its drawbacks: for instance, you must configure all MIDI devices again.

Note: 'Restart with default setup' does not destroy the file lists in the application's menus (such as the Setup submenu — see below): these file lists are not stored in MidiTls.stp but in MidiTls.mru. If you want to clear these file lists, simply use their 'Clear list' operations, or (for a total clearance) remove MidiTls.mru while the application isn't running.

MIDI Tools (macOS)/File (Windows) ➜ Setup:
A submenu from which you can open and save setup ('.stp') files.
A setup file includes nearly all the application's customizable settings, including MIDI I/O device settings and window positions/sizes.
By opening a (previously saved) setup file you can quickly switch from one setup to another. However, opening a setup file involves restarting the application; thus you will lose transient data like unsaved recorded MIDI input messages. So if you only wish to change the window positions/sizes, it's simpler to use the desktop ('.dsk') file facility (see View ➜ Desktop), because this doesn't involve restarting the application.
Tip: you can force MIDI Tools to use a specific setup file via the command line, as follows:
macOS: -s *setupfile*
Windows: /s *setupfile*

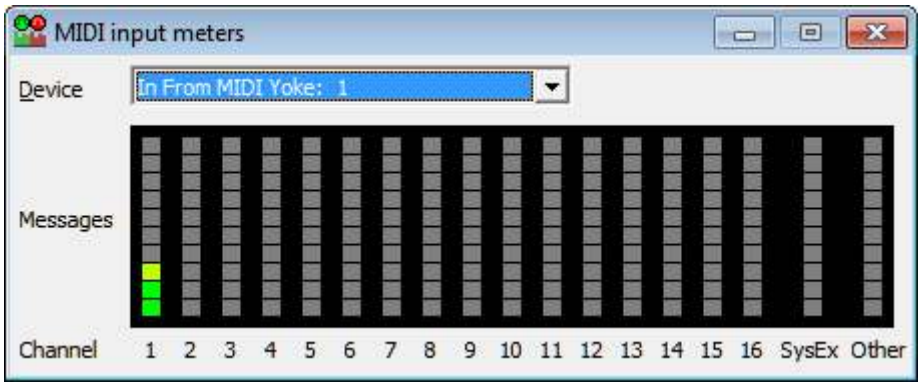
MIDI Tools ➜ Quit (macOS)/File ➜ Exit (Windows):

Terminates MIDI Tools.

Note that on Windows the associated hotkey (Alt+X) works from almost *any* location in the program, not just from the main window. Of course you can also terminate MIDI Tools by clicking on the ✕ icon on the main window's title bar: the same questions are asked. Pressing Alt+F4 also works, but (unlike Alt+X) only from the main window.

View ➜ Input meters:

Opens a window showing the messages received recently from the MIDI input devices, via (logarithmical) LEDs per MIDI channel:



This window can be useful for troubleshooting your MIDI connections.
'DISABLED ' in front of the selected MIDI input device indicates that the device is disabled, so no MIDI messages can currently be received from that device. (You can enable devices in the MIDI devices dialog box.)

View ➜ Input messages:

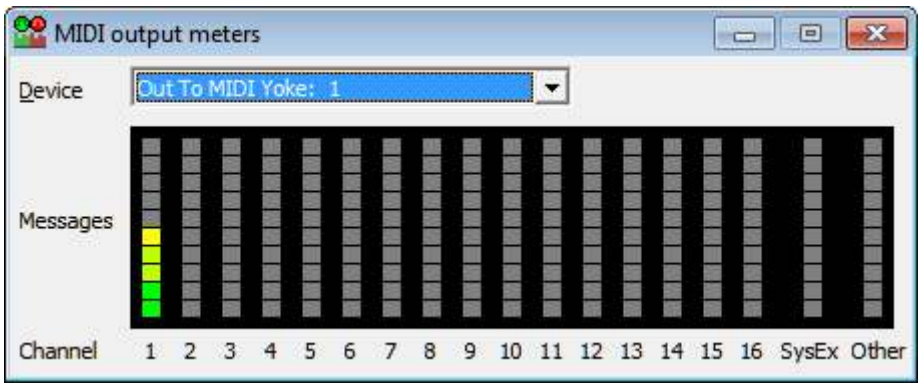
Opens the MIDI input messages window. Here you can record and view messages from the MIDI input devices. See §7 for more information.

View ➜ Mackie monitors ➜ 1/2/3/4:

Opens the indicated Mackie monitor. See §8 for more information.

View ➜ Output meters:

Opens a window showing the messages sent recently to the MIDI output devices, via (logarithmical) LEDs per MIDI channel:



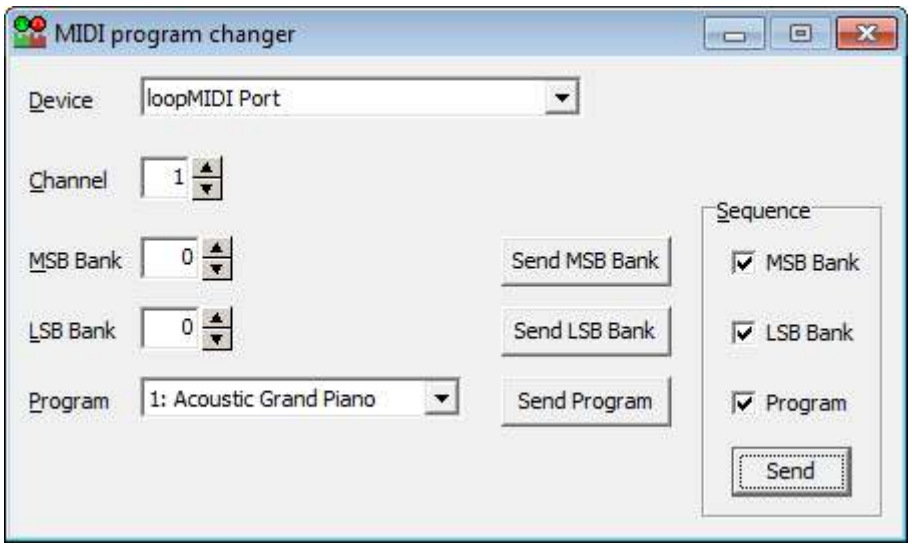
This window can be useful for troubleshooting your MIDI connections.

'DISABLED ' in front of the selected MIDI output device indicates that the device is disabled, so no MIDI messages can currently be sent to that device. (You can enable devices in the MIDI devices dialog box.)

View ➜ Keyboard:
Opens a window containing a virtual MIDI keyboard. See §9 for more information.

View ➜ (N)RPN messages:
Opens a window from which you can send NRPN and RPN message sequences. See §10 for more information.

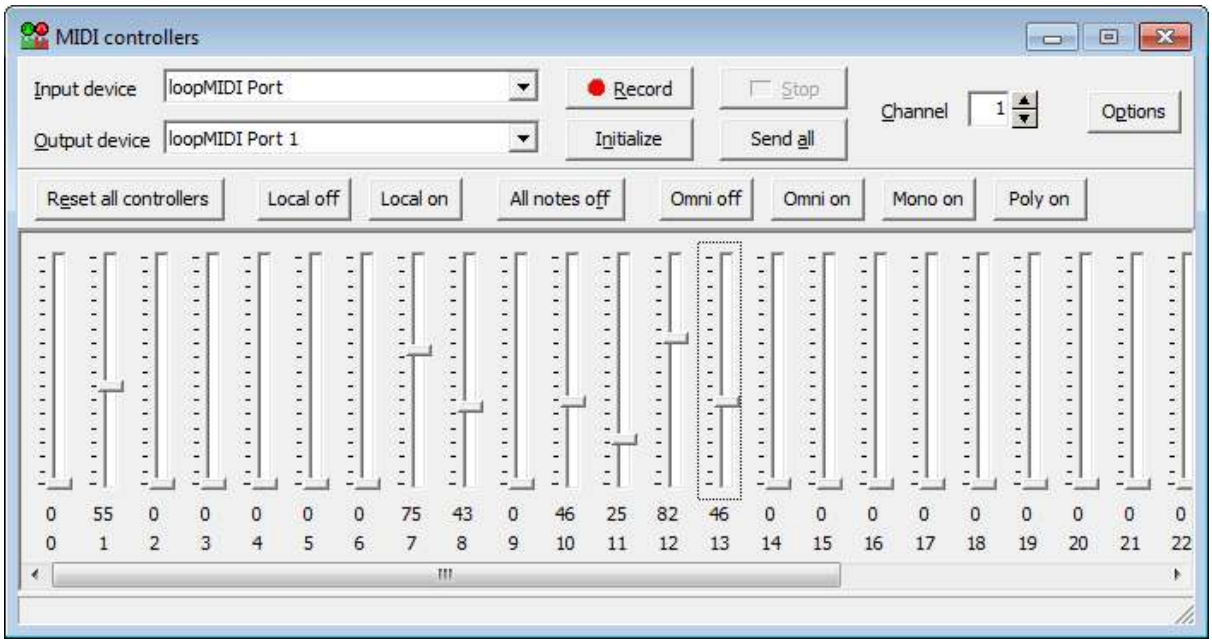
View ➜ Program changer:
Opens a window from which you can send MIDI Program Change messages prefixed by Bank Select MSB/LSB messages, in any combination:



This window is very useful for testing devices that seem unwilling to change programs.

View ➜ System messages:
Opens a window from which you can send MIDI System messages. See §11 for more information.

View ➜ Controllers:
Opens a window in which you can receive and send MIDI Control Change messages:

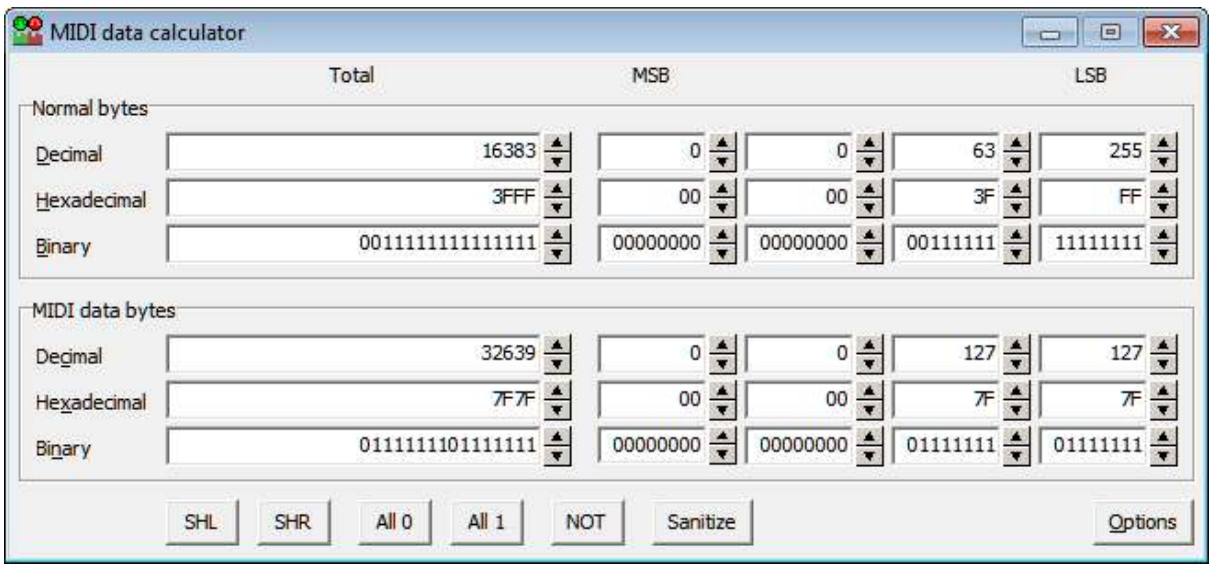


View ➜ System Exclusive messages:

Opens a window in which you can perform many actions related to MIDI System Exclusive (SysEx) messages. See §12 for more information.

View ➜ Data calculator:

Opens a window in which you can convert a 4-byte sequence of 'normal' bytes to its MIDI data counterpart or vice versa. (The MIDI protocol demands that the highest bit ('bit 7') of each MIDI data byte is zero; consequently, if a sequence of two or more MIDI data bytes constitute one number, all bit 7s are 'scrapped' and all more significant bits are shifted to the right.)



Both the normal bytes and the MIDI data bytes are displayed in decimal, hexadecimal and binary form simultaneously. You can also perform simple operations like SHL and SHR. The Sanitize button clears bit 7 of all four MIDI data bytes. Via the Options button you can customize the window's display.

View ➜ Desktop:

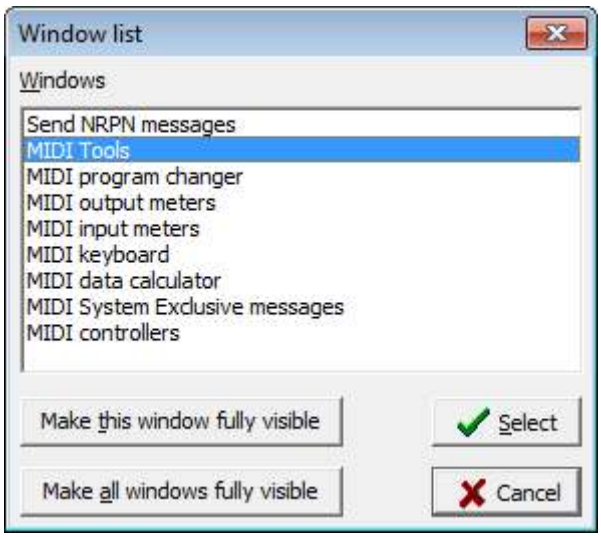
A submenu from which you can open and save desktop ('.dsk') files.
A desktop file contains the positions, sizes and states (hidden/visible/minimized) of all the windows in the application, plus all the display options of the four Mackie monitors. Thus, a desktop file contains a subset of the data in a setup file: see MIDI Tools (macOS)/File (Windows) ➜ Setup.
By opening a (previously saved) desktop file you can quickly switch from one desktop (i.e. layout) to another. Unlike opening a setup file, this does not involve restarting the application.
By default the 'Keep numbers' option is off, so that opening or saving a file promotes the file name to position 1 in the list; when 'Keep numbers' is on, the list stays as it is, which can be useful when you're continually switching between particular desktops.

View ➜ Stay on top:

When this menu item is checked, MIDI Tools' main window stays on top of any other windows belonging to MIDI Tools. On Windows, the main window also stays on top of other applications (except of course those that have the stay-on-top property too).

View ➜ Window list:

Opens a dialog box that allows you to quickly navigate to any open window:



Note that the hotkey (Alt+0) for opening this dialog box works from almost *any* location in the program, not just the main window.

If the highlighted window is partially or completely outside the current monitor(s), you can move it into full view by pressing 'Make this window fully visible'. 'Make all windows fully visible' performs this operation on *all* windows in the list.

Options ➜ MIDI devices:

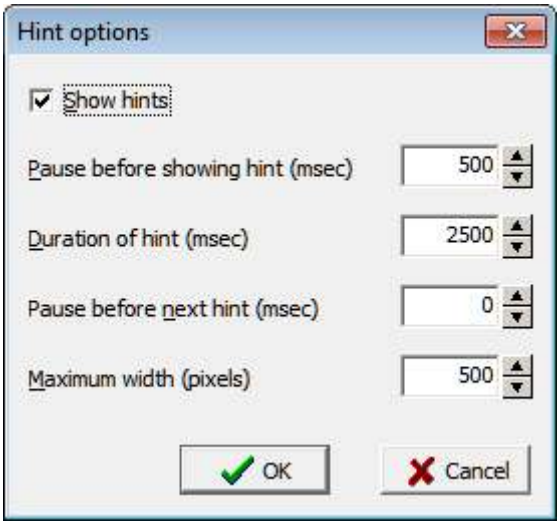
Opens a dialog box in which you can configure the MIDI devices that MIDI Tools monitors. See §5 for more information.

Options ➜ Mackie monitors ➜ 1/2/3/4:

Opens a dialog box in which you can set the selected Mackie monitor's MIDI input port and several display options. See §8 for more information.

Options ➜ Hints:

Opens a dialog box in which you can set options affecting the hints that are displayed when you move the mouse cursor over buttons etc.:



Options ➜ Mouse:

(Note: this menu item actually doesn't occur in the current version of MIDI Tools; however, it does in all the musical hardware editors published by Mountain Utilities.)

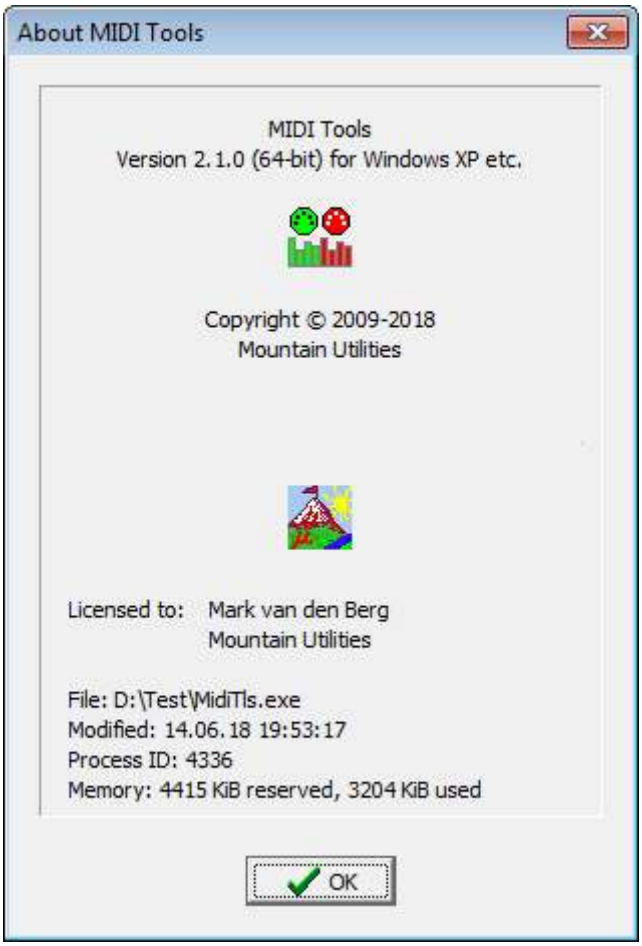
Opens a dialog box in which you can set the way in which the mouse turns the parameter knobs.

Help ➜ Manual:

Opens this manual in the external application associated with the file extension 'pdf'.

MIDI Tools (macOS)/Help (Windows) ➜ About MIDI Tools:

Opens a dialog box containing information on MIDI Tools, such as its version number and memory usage:

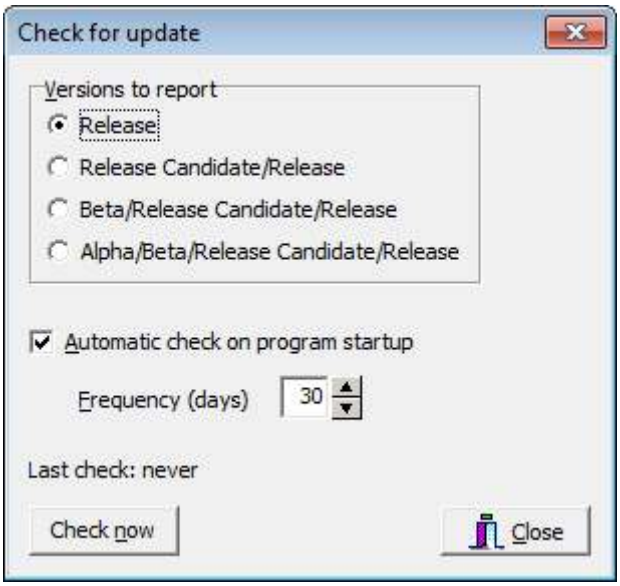


Help ➜ Mountain Utilities web site:

Provided you're connected to the internet, your web browser opens the Mountain Utilities web site, where you can find up-to-date information about MIDI Tools and other Mountain Utilities applications and documents.

Help ➜ Check for update:

Opens a dialog box in which you can set the frequency at which MIDI Tools automatically searches for updates at the Mountain Utilities web site, and which types of updates you are interested in:



If you set ‘Versions to report’ to ‘Release’, you will only be notified about Release versions. ‘Release Candidate/Release’ will also notify you about Release Candidate versions, etcetera.

When an update is available, the program asks you whether you wish to open the program’s web page at the Mountain Utilities site. You can also check for updates manually, by pressing the ‘Check now’ button.

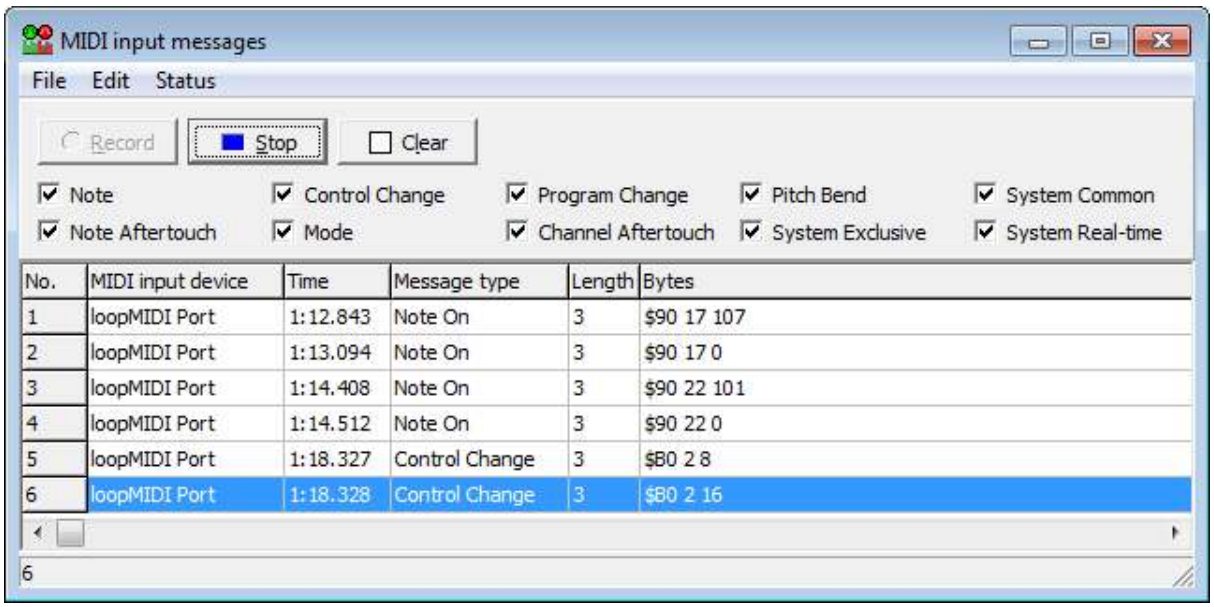
Note: If your firewall catches the program's connection attempt and asks you whether you want to allow this, you can safely say yes: no information identifying you or your computer will be sent to the Mountain Utilities web site.

Help ➜ Donate:

Makes your web browser open the Donate page at the Mountain Utilities web site (https://mountainutilities.eu/donate), at which you can express your appreciation of MIDI Tools and support its further development by making a donation.

7. The MIDI input messages window

The MIDI input messages window is accessible from the main window in two ways: its toolbutton (second from left) and the View pull-down menu ➜ Input messages.



The MIDI input messages window allows you to record and view MIDI messages sent to MIDI Tools from any of the currently *enabled* MIDI input devices, as defined in the MIDI devices dialog box (opened via the main window's Options pull-down menu). Thus, this window is very useful for advanced troubleshooting. You can also save recorded messages to files (in various formats).

All recorded MIDI messages are displayed in the table at the bottom of the window, one per row. The following columns exist:

- No.:
  The sequential number (index) of the message in the table.
  This number is for reference only. It has no further meaning: when you remove a message, the numbers of all subsequent messages simply decrease by one.

- MIDI input device:
  The MIDI input device from which the message was received.

- Time:
  The time at which the message was received, counted from the moment MIDI Tools was started. You can set this column's format in the Edit ➜ Options dialog box (see below).

- Message type:
  The type of the message: Control Change, System Exclusive, etc.

- Length:
  The number of bytes in the message.

- Bytes:
  The bytes of the message. The formatting (hexadecimal, decimal etc.) can be set via the options dialog box, accessed via the Edit pull-down menu.

The menu provides the following operations:

File ➜ Save MIDI file:
Saves the selected (highlighted) MIDI messages to a standard MIDI file ('SMF'). You can load this file in a sequencer program etc.

Two versions of this operation are available via a submenu:

1. 'Times relative to first-saved message':
   The original recording times are maintained, but for convenience a displacement is applied: all messages are saved with their times 'normalized' to the *first* message saved; so the first message saved itself always appears at time 0.

2. 'All times zero':
   All messages are saved with their times set to zero.

Technical notes:

- The MIDI file is in 'format 0', i.e. a single track.

- For convenience, the name of the program ('MIDI Tools') plus its version number is included as the track name. (It completely depends on the receiving program whether you can see this in any way.)

- The file includes a tempo specification of 120 BPM.
  Beware: It seems that when you import a MIDI file into an *existing* Sonar 7 project, Sonar ignores this file tempo of 120 BPM and wrongly interprets the message times according to the existing project's tempo. In the case of a file saved via 'Times relative to first-saved message', this may lead to unwanted stretching, so it's best to only import such a file into a Sonar project having a tempo of 120 BPM. (I haven't tested later Sonar versions yet.)

- MIDI 'running status' is automatically applied, i.e. where possible the status bytes of channel messages are removed.

File ➜ Save binary file:
Saves the bytes of the selected (highlighted) MIDI messages to a binary file. Note that the recording times are *not* saved: you should save to a standard MIDI file for that (see 'Save MIDI file' above).

You can select 'bin', 'syx' or any other extension for the output file, but your choice does not affect the *content* of the output file in any way.

Beware: a *syx* output file is only valid (i.e. usable in a standard way by other programs) if it *only* contains *System Exclusive* MIDI messages. And since MIDI Tools specifically allows you to create a syx file containing only the recorded System Exclusive messages (see 'Save System Exclusive messages' below), the 'save binary file' operation is primarily intended to facilitate further processing by some specialistic computer program expecting a 'flat' sequence of MIDI messages. (Typically this is a program you write yourself!) Note that you can also save MIDI

message bytes to a *text* file (see below), which may or may not be easier for further processing.

File ➜ Save text:
Saves the selected (highlighted) MIDI messages to a text file: the bytes of each message are output on a separate line. The bytes are written in the formats defined in the options dialog box (cf. Edit ➜ Options), so exactly as they are currently being displayed in the Bytes column of the window's table.

You could process the output file in an external text editor, then convert them to a binary file: see 'Convert text file(s) to binary file(s)' below.

File ➜ Save System Exclusive messages:
Saves any selected (highlighted) MIDI *System Exclusive* messages to a syx file.

File ➜ Convert text file(s) to binary file(s):
Converts a text file containing lines of hexadecimal bytes (*without* '$' prefixes) to a binary file. You can select 'bin', 'syx' or any other extension for the output file, but your choice does not affect the *content* of the output file in any way.

This is a somewhat obscure utility that could be applied to a text file created by a 'Save text' operation (see above), possibly edited afterwards via a normal text editor (such as Notepad). Specifically, you can thus convert a text file containing only MIDI System Exclusive messages to a legal syx file.

Edit ➜ Copy to MIDI message clipboard:
Copies the selected (highlighted) MIDI messages to the MIDI messages clipboard. Note that MIDI 'running status' is automatically applied, i.e. where possible the status bytes of channel messages are removed.

Edit ➜ Delete:
Removes the selected (highlighted) recorded MIDI messages.

Edit ➜ Clear:
Removes all recorded MIDI messages.

Edit ➜ Select all:
Selects all recorded MIDI messages.

Edit ➜ Options:
Opens a dialog box in which you can set various options related to the MIDI input messages window:

- Buffer size:
  Sets the number of MIDI messages that can be recorded. The default is 65536; this is also the maximum. Note that lowering this setting removes any existing recorded messages beyond the new buffer size.

- Buffer overflow protocol:
  Determines what happens if the buffer is full (as determined by the 'buffer size' setting) when a MIDI message comes in:

  - Clear:
    The whole table is cleared, and the incoming message is entered at number 1. This is

the default setting.

- Shift:
  The existing message at number 1 is removed from the table, all other messages shift back one position, and the incoming message is added at the bottom.
  Beware: this setting can be very time-consuming.

- Freeze:
  The incoming message is ignored. However, the recording process itself isn't stopped automatically, so when you manually remove one or more recorded messages (e.g. via the Clear button), new messages will be recorded again.

- Stop:
  Recording stops automatically.

- Scroll to new message:
  Determines whether the message table automatically scrolls to any incoming MIDI message.
  'On' is the default, but may result in 'frantic' scrolling when MIDI input is heavy, which may also starve other parts of the program. For instance, the MIDI input and output meter windows may become unable to update their gauges at the required frequency, so that not all incoming messages are displayed. So if you want a quieter display, switch scrolling off.
  Note that the number of recorded MIDI messages is always shown on the status bar at the bottom of the window: this allows you to establish that messages are being recorded even when you have disabled scrolling.

- Time format:
  Determines the time format used in the Time column. Five formats are available: ms, sec.ms, min:sec.ms, hrs:min:sec.ms and days:hrs:min:sec.ms. So e.g. in the sec.msec format you could get '61.000', which would be '1:01.000' in the min:sec.msec format.

- Byte formats:
  Determines the ways in which MIDI message bytes are formatted: this affects both the window's Bytes column and the 'Save text' operation.
  Separate settings are available for 'status' and 'data' bytes in both System Exclusive and non-System Exclusive messages. A byte in a MIDI message is a status byte if it is in the range of $80-$FF (128-255), and a data byte if it is in the range of $00-$7F (0-127).

Status ➜ Record:
Starts the recording process.

Status ➜ Stop:
Stops the recording process.

The panel below the menu contains the following items:

Record/Stop/Clear buttons:
These buttons duplicate the corresponding menu items.

Note/Note aftertouch/etc.:

These checkboxes determine which incoming MIDI messages are recorded. Checked means ‘yes’.

8. The Mackie monitors

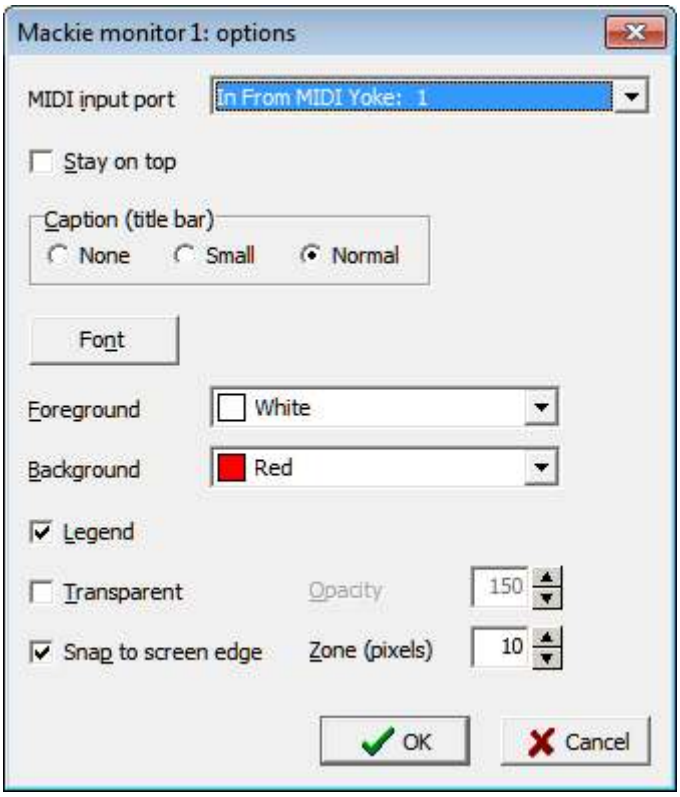
Certain MIDI applications (e.g. Propellerhead's Reason) send display messages to a MIDI output port where they suspect a Mackie Control. (Note: In my experience MIDI applications only send these Mackie display messages in response to parameter changes sent *from* the (pseudo-)Mackie Control *to* the MIDI application, i.e. when you turn a knob on the Mackie Control. At least I've noticed that (curiously enough) Reason does *not* send a display message when you move a fader or knob in Reason itself.)

You can intercept these Mackie display messages and show them in one of four Mackie monitor windows, accessible from the main window's View pull-down menu (➜ Mackie monitors ➜ 1/2/3/4):

Typically you would use one of these Mackie monitors for a MIDI controller (e.g. a Behringer BCF2000 or BCR2000) emulating a Mackie Control.

You set up things as follows: you send the MIDI application's output to a MIDI pipe (virtual MIDI device), you capture this pipe's output in a Mackie monitor in MIDI Tools, and you pass on the MIDI pipe's output to the (pseudo-)Mackie Control's input by means of MIDI Tools' MIDI Thru facility (set via the main window: Options ➜ MIDI devices).

You set the Mackie monitor's MIDI input port and customize its display via a dialog box that you can access via the main window's Options pull-down menu (➜ Mackie monitors ➜ 1/2/3/4) or by right-clicking on the monitor window itself:



This dialog box offers three options for ‘Caption (title bar)’:

- None:
  The monitor window has no border and no title bar. So you can’t move the window around the screen via the title bar, but you can do so by keeping the left mouse button down anywhere in the window itself.
  If ‘Stay on top’ is checked, the Windows taskbar shows the monitor window on a separate button, but you can’t use that button to minimize (i.e. temporarily hide) the window.

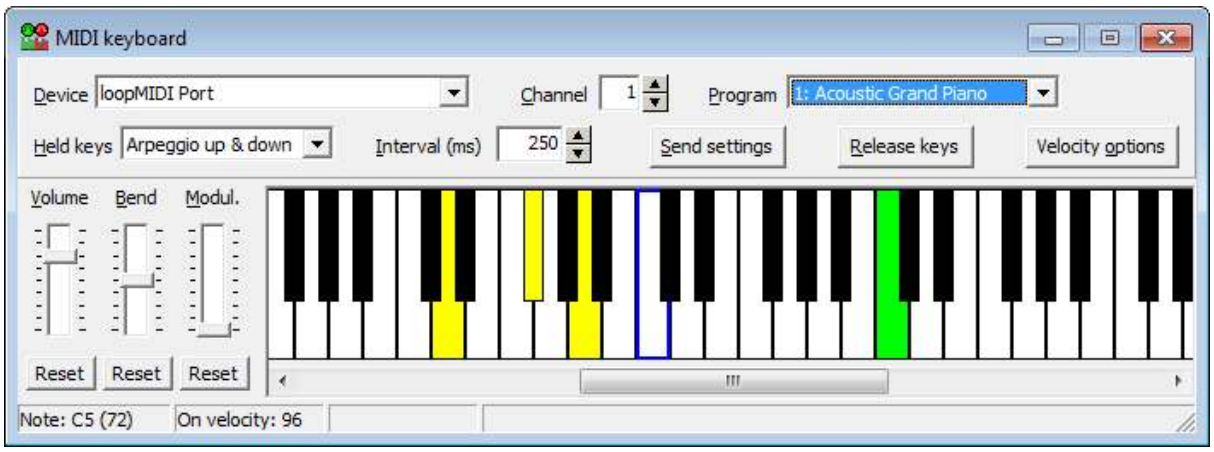
- Small:
  This setting is only meaningful on Windows (on macOS the result is the same as for Normal – see below): the monitor window is a ‘tool window’, having a border and a small title bar *without* a minimize icon.
  If ‘Stay on top’ is checked, the Windows taskbar does *not* show the monitor window on a separate button.
  So there is no way to minimize (i.e. temporarily hide) the window.

- Normal (this is the default setting):
  The monitor window has a border and a normal title bar (including a minimize icon).
  If ‘Stay on top’ is checked, the Windows taskbar shows the monitor window on a separate button: when you click that button, the monitor window toggles between being shown and being minimized, independently of the application’s main window.

Note that on macOS the ‘Stay on top’ setting doesn’t have any effect.

9. The MIDI keyboard

The MIDI keyboard is accessible from the main window in two ways: its toolbutton and the View pull-down menu ➜ Keyboard.



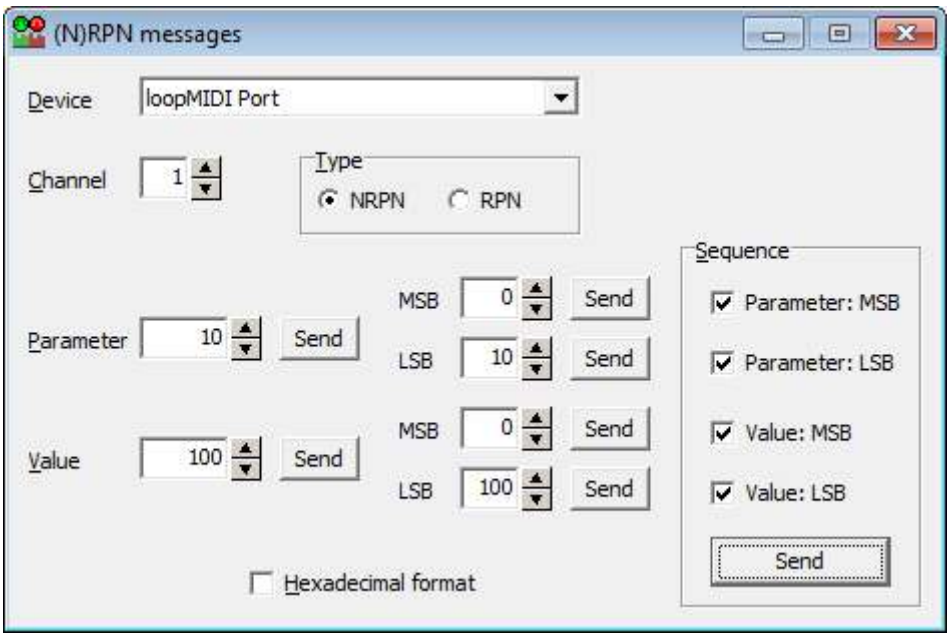
From the MIDI keyboard you can send note messages (and several related other messages) to any MIDI output device (e.g. a synthesizer):

- Note names displayed on the status bar follow the Roland octave numbering protocol.
- On the keyboard, Middle C (#60, C4) has a blue border.
- Pressing the 'Send settings' button sends the selected program, volume, pitch bend and modulation to the selected MIDI output device.
- To play an *individual* note, *left*-click on a key: the key turns green for as long as you keep the mouse-button pressed.
- To *hold* a note, *right*-click on a key: the key turns yellow, until you click it again (or clear all held keys).
- The 'Held keys' drop-down box determines how the held keys are interpreted: they can be played as a chord (i.e. simultaneously) or as arpeggios.
- 'Interval (ms)' determines the chord or arpeggio time interval.
- Pressing the 'Release keys' button clears all currently held keys.
- If you hold the Shift key while left-clicking or right-clicking, all currently held notes are cleared before the newly selected note is played or held. (In other words: it's as if you've pressed the 'Release keys' button.)
- Beware: at least on Windows (I'm not sure about macOS) the chord/arpeggio timing is not very accurate, because the 'simple' Windows timer is used (rather than the multi-media timer, which is more exact but also puts a larger strain on the computer). In particular, performing actions like opening a listbox may stall the timer. In other words: don't try to use the chord/arpeggio modes

for actual music production!

10. The (N)RPN messages window

The (N)RPN messages window is accessible from the main window's View pull-down menu ➜ '(N)RPN messages'.



From the (N)RPN messages window you can send NRPN and RPN message sequences:

A complete NRPN or RPN message sequence sends a parameter (i.e. a number) followed by a data value for that parameter. Non-registered parameter numbers are specific to individual receiving MIDI devices; registered parameter numbers are defined in the MIDI specification, but are rarely implemented in actual devices.

Each parameter or value is a 14-bit value in the range 0-16383, divided into the MSB (bits 7-13) and the LSB (bits 0-6); the MSB must be sent before the LSB. Thus, a complete NRPN or RPN message sequence consists of four messages in fixed order.

Each of these messages is actually a MIDI Control Change message: the first byte is $B*c* (where *c* stands for the MIDI channel (1-16) minus 1), the second byte is the 'controller' number (e.g. 99 for the MSB of an NRPN parameter), and the third byte is the actual MSB or LSB of the parameter or value:

NRPN:
1. $B*c* 99(=$63) *ParameterMSB*
2. $B*c* 98(=$62) *ParameterLSB*
3. $B*c* 6(=$06) *ValueMSB*
4. $B*c* 38(=$26) *ValueLSB*

RPN:
1. $B*c* 101(=$65) *ParameterMSB*
2. $B*c* 100(=$64) *ParameterLSB*
3. $B*c* 6(=$06) *ValueMSB*
4. $B*c* 38(=$26) *ValueLSB*

However, messages 1 and/or 2 can sometimes be omitted from these sequences:

- A particular receiving device may ignore either *ParameterMSB* or *ParameterLSB* altogether, so that message 1 or 2 respectively may be omitted.
- Message 1 may be omitted if *ParameterMSB* equals the most recently sent *ParameterMSB*.
- If message 1 is omitted, message 2 may be omitted too if *ParameterLSB* equals the most recently sent *ParameterLSB*.

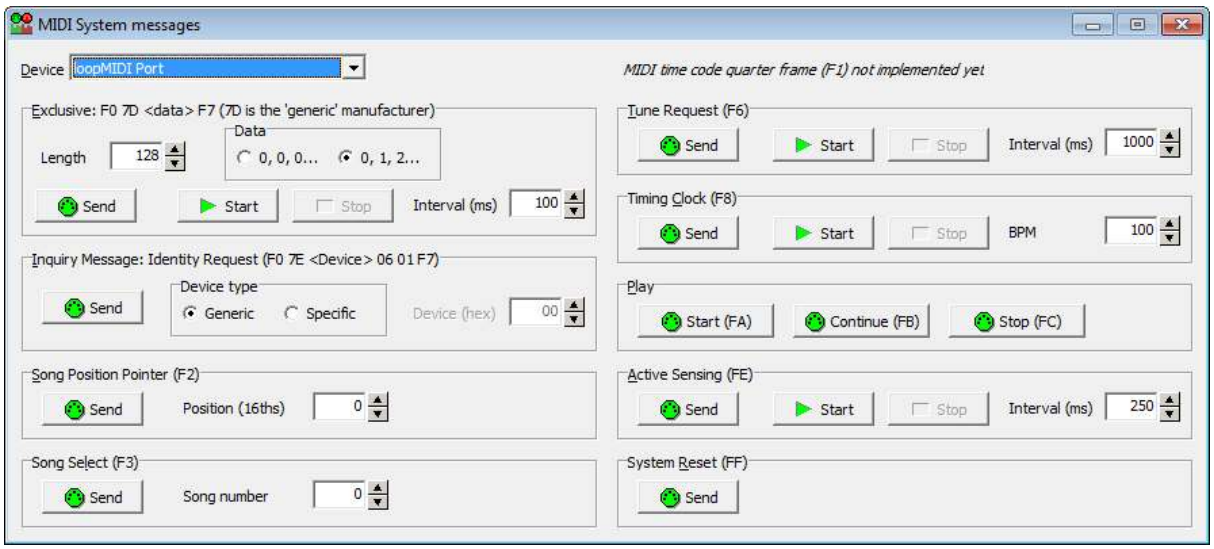
Similarly, message 3 or message 4 can sometimes be omitted:

- A particular receiving device may ignore either *ValueMSB* or *ValueLSB* (possibly just for particular parameters), so that message 3 or 4 respectively may be omitted.
- Message 3 may be omitted if *ValueMSB* equals the most recently sent *ValueMSB*.

Clearly, since MIDI has a limited bandwidth, it's a good thing that certain messages can be omitted as described above. However, as will be clear from the above definitions, these omissions can cause confusion, and devices may respond to them in unexpected, idiosyncratic ways. Therefore the (N)RPN messages window allows you to send individual messages or any combination of messages, so that you can test the actual behavior of the receiving device.

11. The MIDI System messages window

The MIDI System messages window is accessible from the main window's View pull-down menu ➜ 'System messages'.



From this window you can send MIDI System Exclusive (SysEx), System Common and System Real-time messages – presumably mostly for testing purposes.

Some remarks:

System Exclusive:

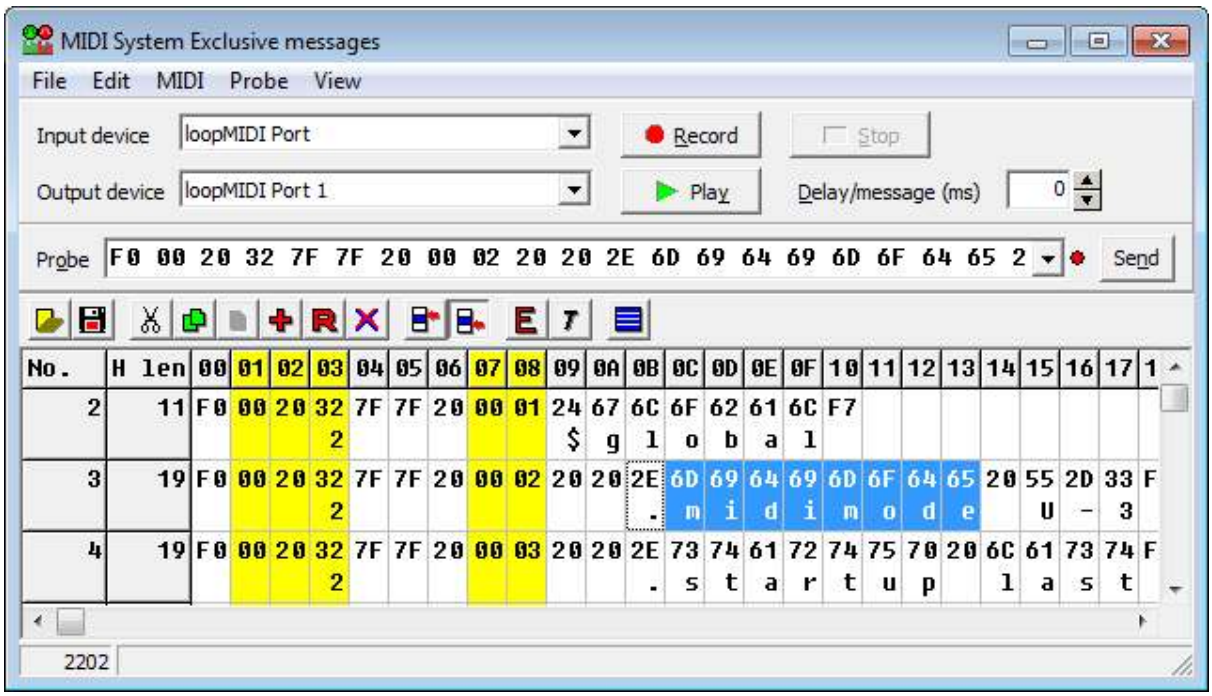
- Each SysEx message generated here contains 7Dh as Manufacturer, which indicates non-commercial, generic use. Thus, these messages are normally ignored by all receiving MIDI devices.

  The purpose of these messages is to test SysEx signal flow, in particular the behavior of devices (drivers, sound cards, devices with 'MIDI Merge' facilities, etc.) through which these messages pass. For instance, some devices handle SysEx messages beyond certain lengths incorrectly. (E.g. the Behringer BCF2000 and BCR2000 MIDI Controllers make garbage of any SysEx message longer than 1019 bytes sent from/to their MIDI IN/OUT sockets via their USB connections.)
- The 'Length' field indicates the *total* length of the SysEx message, i.e. F0 7D <*data bytes*> F7.
- The 'Data' field determines whether the data bytes are all 0, or increment by 1 from 0 to 127 (then start from 0 again).
- To generate your own SysEx messages, use the 'MIDI System Exclusive messages' window instead; see §12.

Tune Request, Timing Clock, Active Sensing:
Beware: on Windows, the timing of the sequences of Tune Request, Timing Clock and Active Sensing messages (generated via the Start buttons) is not very accurate, because the 'simple' Windows timer is used. Cf. a remark in §9. (I haven't yet checked how things are for macOS.)

12. The MIDI System Exclusive messages window

The MIDI System Exclusive messages window is accessible from the main window in two ways: the ‘SYX’ toolbutton and the View pull-down menu ➜ ‘System Exclusive messages’.



In this window you can create and edit SysEx messages, but also insert SysEx messages from syx files, save messages to syx files, and record and play SysEx messages.

Furthermore, you can extract the SysEx messages from a standard MIDI (‘.mid’) file and save them to syx files. One syx file is created for each MIDI track (in the mid file) containing one or more SysEx messages.

The window contains three panels:

1. The MIDI panel.
   Here you can select the MIDI input and output devices.
   When you press Record, incoming messages will be recorded into the grid below the toolbar.
   When you press Play, the messages selected in the grid are played, with the pause between messages being determined by ‘Delay/message (ms)’.
   Note that ‘Delay/message (ms)’ is also applied when you press Send in the probe panel if the probe line contains more than one message.

2. The probe panel.
   This panel contains an editbox in which you can type one or more SysEx messages as a sequence of bytes (in bare hexadecimal format, separated by spaces), which you can send to the selected MIDI output device.
   You can also save and reopen the history list of the editbox to/from text (‘.txt’) files.

3. The toolbar.
   This contains a number of buttons performing actions related to the grid below.

The grid below the toolbar contains a sequence of messages (recorded/opened/edited).
You can edit the data in this grid in two ways: per byte or per message.

Both the probe panel and the grid allow you to correct the checksums of Roland DT1 and RQ1 messages, either automatically or on demand:

- If 'Auto-correct Roland checksum(s)' in the Edit pull-down menu has been ticked, the 'Edit byte(s)' and 'Edit message(s)' operations automatically correct any invalid Roland checksums resulting from your edits to the grid.
- If 'Auto-correct Roland checksum(s)' in the Probe pull-down menu has been ticked, the message(s) in the probe editbox are automatically corrected when you execute Send.
- When you execute 'Correct Roland checksum(s)' from the Edit pull-down menu, any invalid checksums in the selected messages in the grid get corrected.
- When you execute 'Correct Roland checksum(s)' from the Probe pull-down menu, any invalid checksums in the probe editbox get corrected.

From the View pull-down menu you can customize the window in several ways:

- You can hide any of the three panels (MIDI, probe, toolbar).
- You can include or exclude the horizontal and vertical grid lines, separately for the header and leader cells (gray in the screenshot above) and the (white) data cells. (However, on macOS the grid lines in the header and leader cells don't get hidden. This is probably due a bug in the programming library for macOS used by MIDI Tools, so at the moment I can't fix this easily.)
- You can display the message lengths in decimal or hexadecimal format.
- You can display the message bytes in various formats: decimal, hexadecimal (the default), binary and character (insofar as the bytes are in the ASCII range). Note that you can select as many formats simultaneously as you like: each format is displayed on a separate line.
- If 'Select row' is on, you can only select (and edit) whole lines (messages); if it is off, you can select (and edit) blocks of bytes. Note that in the latter case certain operations (like Play and 'Correct Roland checksum(s)') still use the *whole* message(s) of the selected bytes.
- You can highlight/unhighlight any column by right-clicking anywhere in the column or by executing View ➜ 'Highlight selected column' from the menu. (Cf. the yellow columns in the image above.) You can even customize the highlight color.

13. Using the computer keyboard

MIDI Tools' user interface uses mostly standard widgets (buttons, checkboxes, pull-down boxes etc.). This means that it may sometimes be easier to use the keyboard instead of the mouse for particular operations.

For Windows, the following standard keystrokes are worth mentioning:

Control	Key(s)	Action
Any	Tab	Select the next control
	Shift+Tab	Select the previous control
Checkbox	Space	Toggle the setting on/off
Pull-down box	Left/Up arrow	Select the previous item
	Right/Down arrow	Select the next item
	Home	Select the first item
	End	Select the last item
	Alt+Up/Down arrow	Open/close the pull-down list

And here are some important keystrokes and mouse-clicks for knobs (*note: actually knobs don't occur in the current version of MIDI Tools; however, they do in all the editors for music hardware published by Mountain Utilities*):

Key(s)/mouse click	Action
Left arrow	Decrease the value by 1
Right arrow	Increase the value by 1
Ctrl+Left arrow	Decrease the value by a 'big' amount (often 10)
Ctrl+Right arrow	Increase the value by a 'big' amount (often 10)
Home	Select the lowest value
End	Select the highest value
Alt+Enter	Open a dialog box in which you can type a new value (this only works for purely numerical knobs, i.e. knobs without items like 'Off')
Left click (caption/value)	Select the knob under the mouse
Left click (actual knob)	Set the value as indicated by the mouse position
Right click	Change the value by a 'big' amount (often 10) in the direction of the mouse

14. Known problems

MIDI Thru (Windows only):

MIDI Tools' MIDI Thru feature only passes on *short* MIDI messages, i.e. any message *except* SysEx (System Exclusive).

This is because MIDI Tools achieves its MIDI Thru feature by simply calling the midiConnect function in Windows' MMSystem library: basically Windows handles all Thru traffic behind MIDI Tools' back, but unfortunately midiConnect doesn't pass on SysEx messages. (Incidentally, the Huskervu utility's MIDI Thru feature doesn't pass on SysEx messages either, so it probably uses midiConnect as well!)

I may try to find a work-around for this in a future version of MIDI Tools. In the meantime you should use MIDI-OX if you need to pass on SysEx messages via a MIDI Thru connection. (Apparently MIDI-OX doesn't use midiConnect, but handles all MIDI Thru traffic manually, which might actually be marginally slower than midiConnect, for non-SysEx messages that is...)

USB MIDI ports:

While MIDI Tools is running, connecting or disconnecting a MIDI device via its USB cable must be avoided, since it may lead to nasty error messages; instead, you must exit and restart MIDI Tools manually. I've been working on a fix, but I don't know if and when this will be made available.

Window widths:

If the screen dimensions are too small, big windows of *fixed* size can get cut off. Normally you're safe with a screen of 1024 × 768 pixels, but you can run into problems when you decrease the screen size of a virtual machine running MIDI Tools.

Alternative DPI settings:

Nearly all screen elements scale correctly under alternative DPI settings. However, the bitmaps used in the pull-down menus *don't* scale, which leads to rather cramped-looking pull-down menus at high enlargement DPI settings, because the heights of the menu items follow these bitmaps instead of the menu items' *names* (which *do* scale).

macOS only:

Several shortcomings of the program's GUI (graphical user interface), such as:

- Scrollbars around grids don't go away once they are there: once you've made such a scrollbar appear by making the window smaller, it remains, even when you re-enlarge the window.
- Numbers in editboxes don't align to the right but to the left. (Even worse, a long number may initially be partly hidden behind the left edge of the editbox.)

All these problems are caused by bugs in the GUI library the program uses on macOS, so at the moment there's not much I can do.