TRANSMIDIFIER

USER GUIDE

Version 1.0 Beta

The only warranties for Be Wary Software products and services are set forth in the express warranty statements accompanying such products and services. Nothing herein should be construed as constituting an additional warranty. Be Wary Software shall not be liable for technical or editorial errors or omissions contained herein.

Be Wary Software assumes no responsibility for the use or reliability of its software on equipment that is not furnished by Be Wary Software.

Be Wary Software
1742 Glendon Avenue, Unit A
Los Angeles, CA 90024
USA

http://www.bewaryprods.com/software/

Be Wary Software supports lawful use of technology and does not endorse or encourage the use of our products for purposes other than those permitted by copyright law.

The information in this document is subject to change without notice.

TABLE OF CONTENTS

INTRODUCTION

Thank you for downloading TransMIDIfier! TransMIDIfier was originally created to enable switching among many MIDI instrument patches from a single MIDI track, no matter the instrument, sequencer or digital audio workstation (DAW). During development, many more possibilities were discovered and implemented. The result is an application that not only makes a consistent interface to all your MIDI instruments possible, but also enables powerful customization and combinatory capabilities not easily done with other methods.

SYSTEM REQUIREMENTS

Windows:

- Windows XP or later, 32- or 64-bit.
- At least one MIDI input and output device.

Mac:

- Mac OS X 10.6.8 (Snow Leopard) or later.
- At least one MIDI input and output device.

INSTALLATION

Windows:

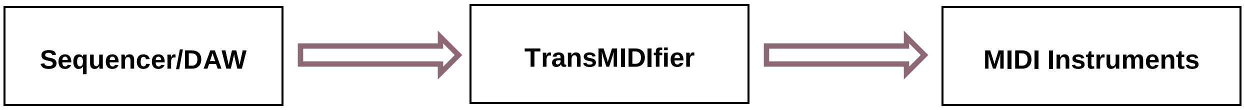
- Save the .zip file to your system, then double-click on it to reveal its contents.
- Copy the entire TransMIDIfier folder to your Windows "Program Files" folder.
- Double-click TransMIDIfier.exe (inside the TransMIDIfier folder) to run the application.
- If you like, create a shortcut to TransMIDIfier.exe by right-clicking on it and selecting "Create Shortcut." The created shortcut can then be moved to your Desktop or wherever you prefer.

Mac:

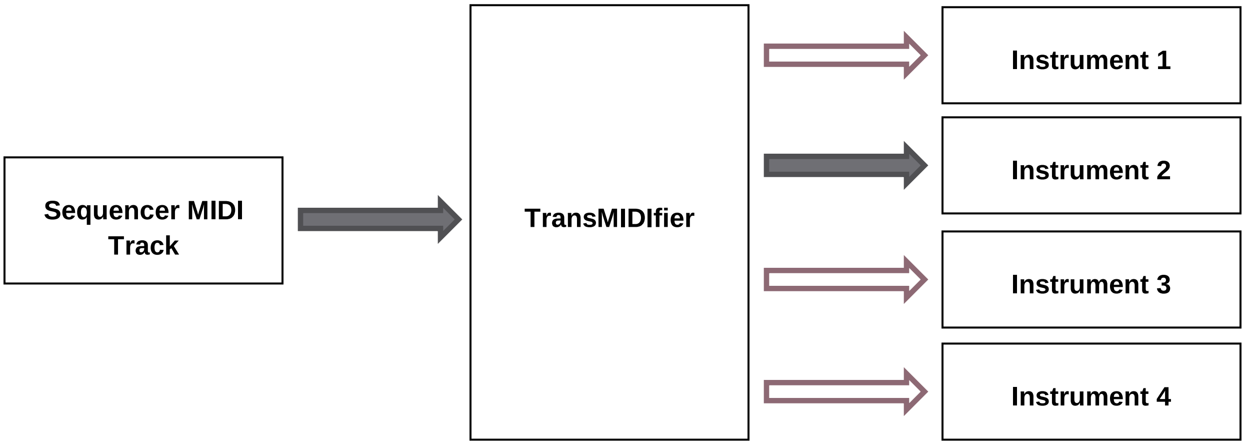
- Save the .zip file to your system, then double-click on it to reveal its contents.
- Copy TransMIDIfier.app into your Applications folder.
- Double click TransMIDIfier.app in your Applications folder to run the application.
- While running, right-click on the TransMIDIfier icon in the system dock and select "Keep In Dock" for quick access in the future.

BASIC CONCEPTS

TransMIDIfier functions as a "middle man" between the sequencer and the MIDI instruments. MIDI data is sent from the master sequencer/DAW to TransMIDIfier, where it is interpreted and perhaps massaged before being passed along to your instruments:



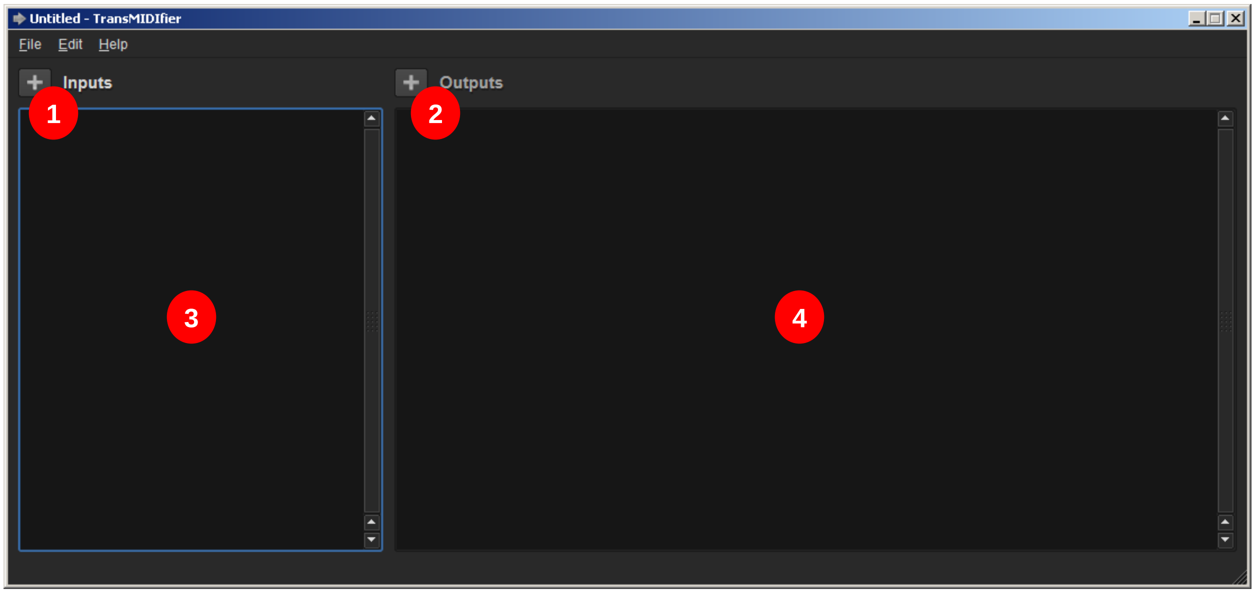
At the core of TransMIDIfier is the ability to receive MIDI data from a particular port/channel and selectively reroute it to many different output ports/channels. As a MIDI track in a sequencer/DAW typically outputs to one MIDI output port on a specific channel, and an instrument is typically addressed at a certain MIDI port on a specific channel, a 1-to-1, MIDI track to virtual instrument relationship is common. But with a virtual instrument such as a cello section, there are many articulations (playing styles) possible, such as sustain, legato, tremolo, staccato, spiccato, pizzicato, harmonics, and trills. Having to create one MIDI track in the sequencer for each and every articulation can lead to lots and lots of tracks. Lots and lots of MIDI tracks means a lot of time spent scrolling around in the sequencer, and difficulty doing things like doubling parts. With TransMIDIfier, one MIDI track can target all possible articulations.



Targeting many different articulations from one MIDI track is just the beginning. TransMIDIfier also allows selectively combining arbitrary instruments together and applying different transformations to the MIDI data being sent to each. This makes it possible to have a consistent interface to all of your virtual instruments, regardless of manufacturer, with a streamlined MIDI track count. The next sections go into more detail on all aspects of this functionality.

Note: screen shots in proceeding sections are taken from the Windows version of the application, but the Macintosh version is for all intents and purposes identical. Mac users, do not be afraid.

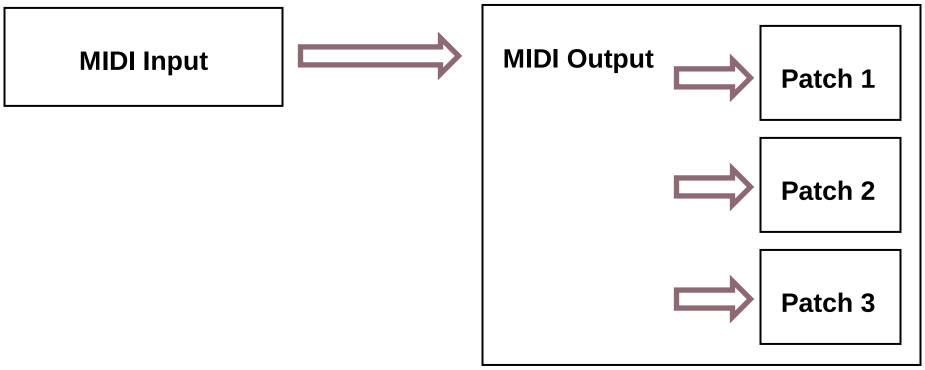
THE TRANSMIDIFIER MAIN WINDOW



1. **Add Input** button. Clicking this button adds a new MIDI input to the setup.
2. **Add Output** button. Clicking this button adds a new MIDI output to the active MIDI input.
3. Input stack. All of the setup's MIDI inputs will be stacked up in this area.
4. Output stack. All of the setup's MIDI outputs will be stacked up in this area.

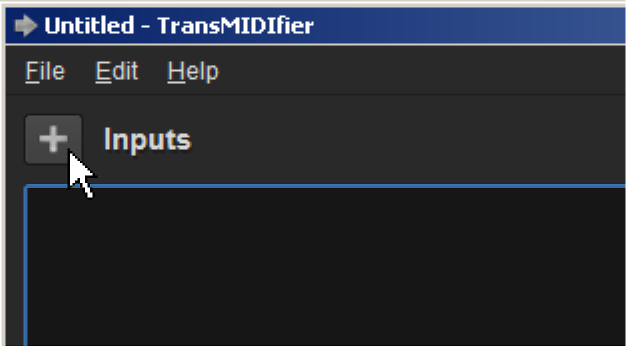
INPUTS, OUTPUTS, PATCHES

TransMIDIfier setups (like "sessions" or "documents") contain many MIDI inputs. The MIDI input is just a named selection of MIDI input port and channel – the place where the incoming MIDI data will be received. Each MIDI input feeds its MIDI data to at least one MIDI output. Typically many MIDI outputs will be associated with a MIDI input, however only one MIDI output for each input is "active" at a time. MIDI data received by the input is passed only to the active MIDI output. The active MIDI output is changed by either program change messages ("PC#") or keyswitches ("Key"). Each MIDI output contains one or many patches. A patch should be thought of as "an individually addressable instrument." The active MIDI output passes its incoming MIDI data to all of its patches simultaneously.

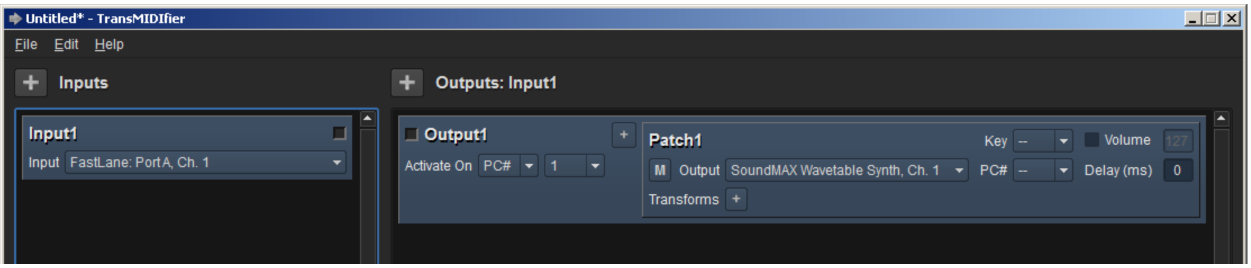


MIDI INPUTS

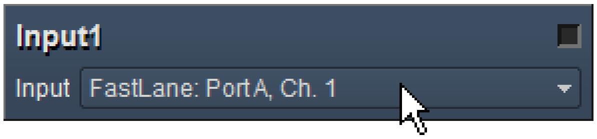
Click the **Add Input** button to create a new MIDI input. You can add as many inputs as you like.

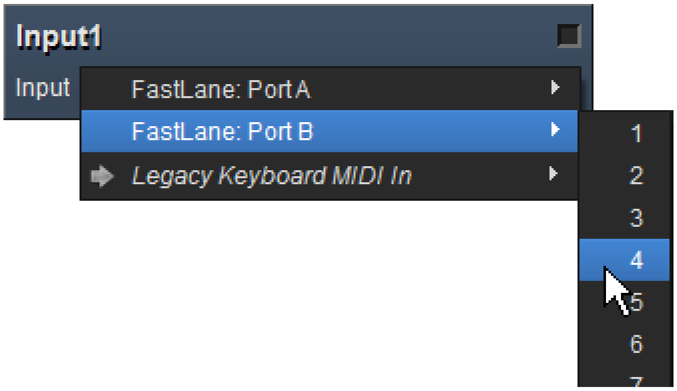


Your newly-created MIDI input will also create its first, default MIDI output, which will contain its default patch.

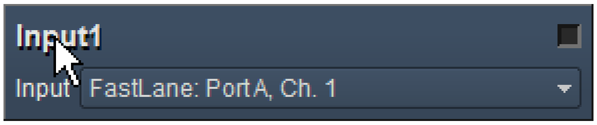


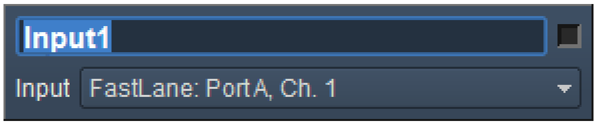
Set the MIDI input's port and channel.



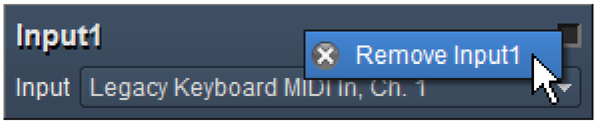


Rename the MIDI input by double-clicking on the current name and typing in a new name.

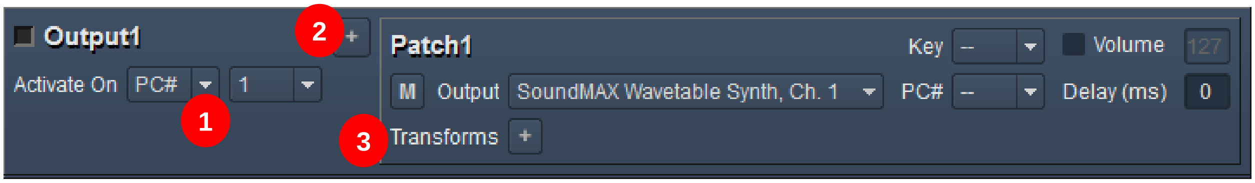




Any MIDI input can be removed by right-clicking on it and selecting “Remove.”

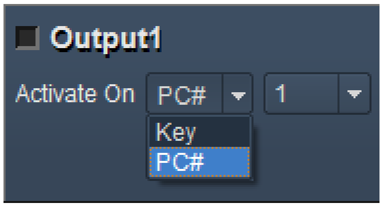


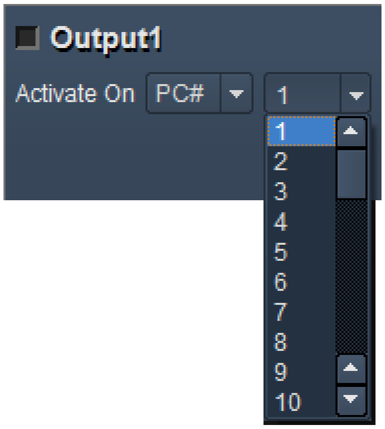
MIDI OUTPUTS

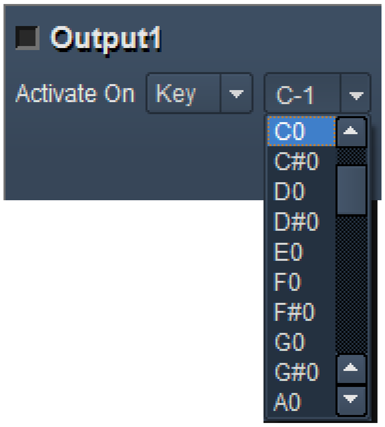


1. Output activation method selector. Choose the method with which the output will become active – either program change (“PC#”) and index or keyswitch (“Key”) and note name.
2. **Add Patch** button. Click this button to add an additional patch to this output.
3. Patch stack. All patches contained within the output will be stacked up here.

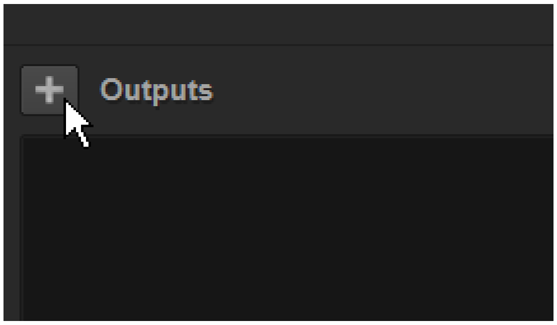
As with MIDI inputs, outputs can be renamed by double-clicking on the current name and typing in a new name. To activate the output via program change, select “PC#” and select the desired program number (1-128). To activate on keyswitch, select “Key” and choose the note name (C-1 to G9).

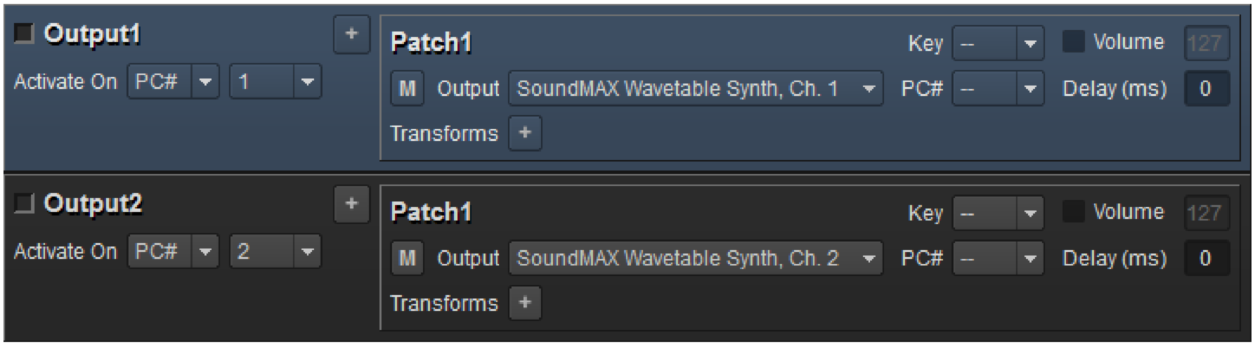




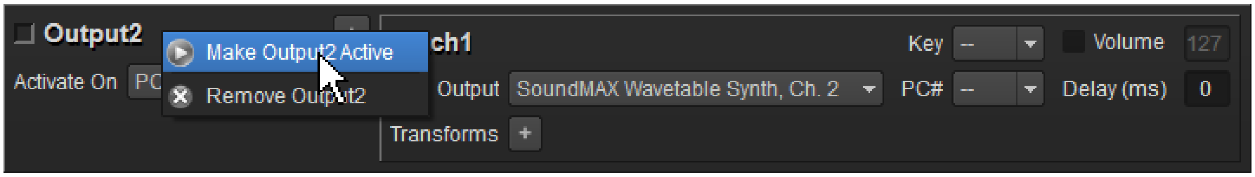


So far we only have one output for this input, which isn’t very interesting, as all MIDI input will simply be passed to one output. Click the **Add Output** button to add a second output:

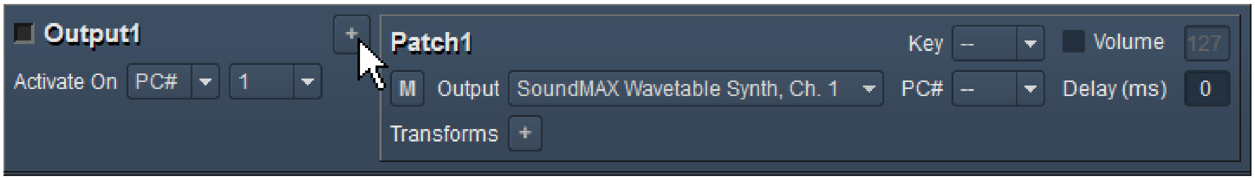


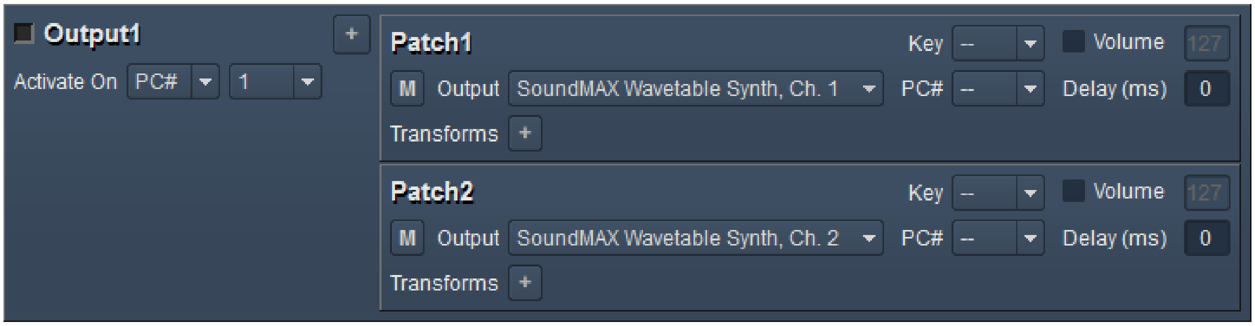

Output1
Activate On PC# 1
Patch1 Key -- Volume 127
Output SoundMAX Wavetable Synth, Ch. 1 PC# -- Delay (ms) 0
Output2
Activate On PC# 2
Patch1 Key -- Volume 127
Output SoundMAX Wavetable Synth, Ch. 2 PC# -- Delay (ms) 0

Now we have two outputs, with PC1 activating the first and PC2 activating the second. The first output, Output1, is currently active, which is indicated by the blue background color. While the setup is running, when this MIDI output receives a PC2 message, Output2 will become active and all subsequent MIDI messages will be sent to Output2, and processed by Output2's patches. When a PC1 message is received, Output1 will be made active again and subsequent MIDI messages will be sent to Output1, and processed by Output1's patches. And so on. A MIDI output can also be activated by double-clicking on it (somewhere other than the name) or by right-clicking on the output and selecting **Make <output name> Active** from the context menu:

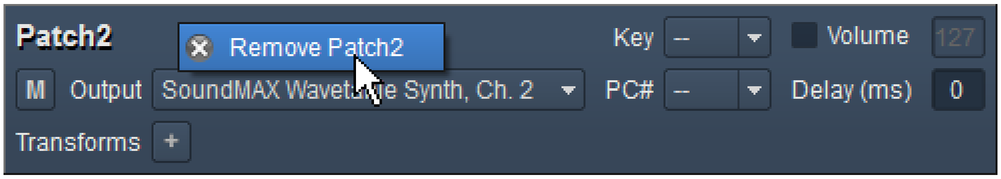

Output2
Key -- Volume 127
Output SoundMAX Wavetable Synth, Ch. 2 PC# -- Delay (ms) 0

Outputs can also be removed via the right-click context menu, just like inputs. Also like inputs, you can create as many outputs as you like. Click the **Add Patch** button to add a second patch to the output:

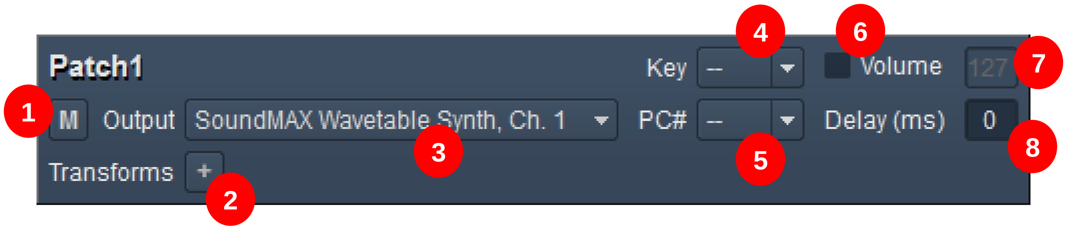

Output1
Activate On PC# 1
Patch1 Key -- Volume 127
Output SoundMAX Wavetable Synth, Ch. 1 PC# -- Delay (ms) 0


Output1
Activate On PC# 1
Patch1 Key -- Volume 127
Output SoundMAX Wavetable Synth, Ch. 1 PC# -- Delay (ms) 0
Patch2 Key -- Volume 127
Output SoundMAX Wavetable Synth, Ch. 2 PC# -- Delay (ms) 0

Just like inputs and outputs, patches can be renamed by double-clicking on the current name, and they can be removed via the right-click context menu:



PATCHES

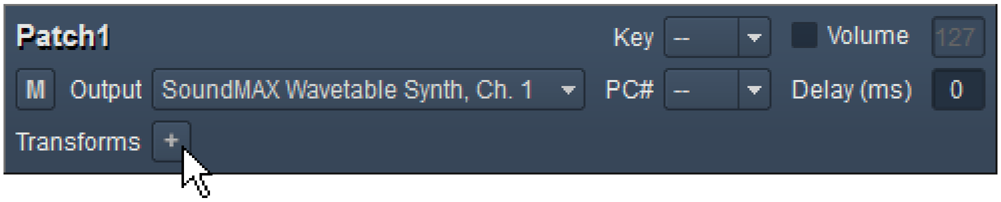


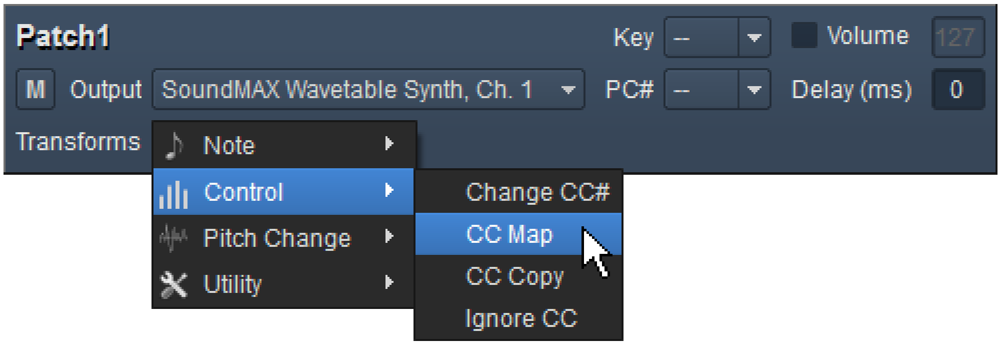
1. Mute toggle button.
2. **Add Transform** button.
3. MIDI output port/channel selector.
4. Initial keyswitch selector.
5. Initial program change selector.
6. Initial volume enable checkbox.
7. Initial volume edit box.
8. Delay edit box.

The patch is the gateway through which MIDI data is sent to a specific instrument. An instrument is targeted by the patch with the output port/channel selector, similar to how an input selects its input port/channel. The patch has two additional addressing options with the initial keyswitch and program change selectors. If the instrument the patch desires to target lives within an already keyswitched instrument or bank of patches addressable by program change, the “Key” and “PC#” selectors can be used to specify the exact sound or articulation desired. When the patch’s parent output is activated, all of its patches’ initialization activities will be performed before any MIDI messages are processed. These initialization activities include the initial keyswitch and program change, if specified, plus an optional initial volume. When the initial volume checkbox is checked, the specified volume value, 0-127, will be sent to the specified output (as a CC7 message, after any initial program change and keyswitch). The delay value is used to delay the processing of all MIDI data by the specified number of milliseconds (500ms maximum). A patch can be muted (and unmuted) by clicking the mute toggle button. When muted, a patch does not process any of MIDI data passed to it. Finally, each patch may contain a series of transforms. Transforms are operations performed on the MIDI data before it is sent out to the MIDI output ports. Transforms are covered in greater detail in the following section.

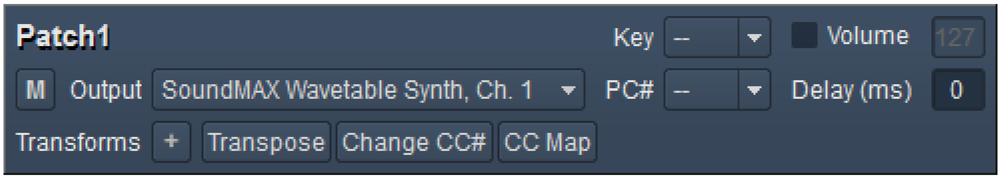
TRANSFORMS

Transforms are operations performed on the MIDI data before it is sent out to the MIDI output ports. The different transform possibilities are listed below. Each patch can have zero or many transforms. A new transform is added to the patch by clicking the **Add Transform** button and selecting a transform from the menu:





After they're added to a patch, transforms can be seen along the bottom of the patch panel:

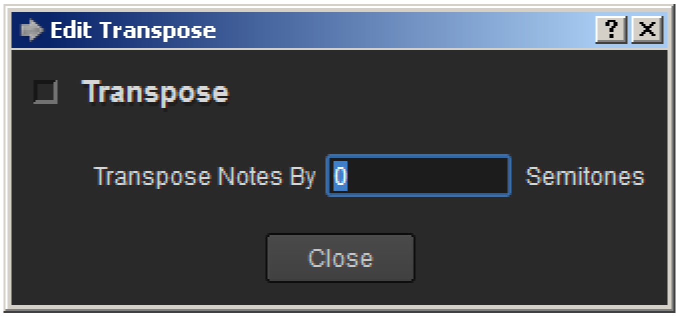


To edit a transform, double click on the transform indicator, which will bring up the appropriate transform editor dialog. The available transforms and their editors are described in the following sections.

NOTE TRANSFORMS

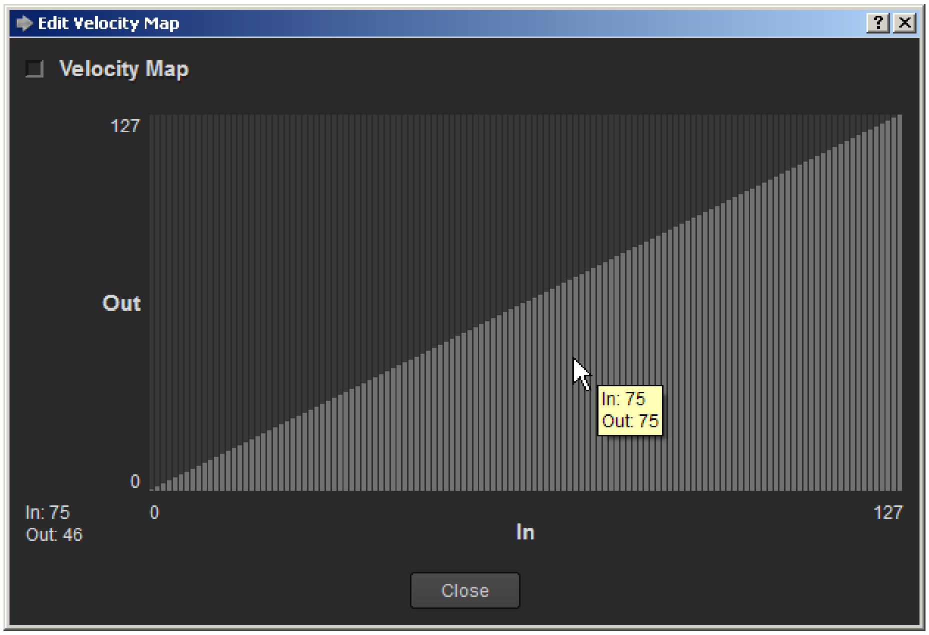
Note transforms either operate on note messages or are derived from note messages. The various note transforms are described below.

TRANSPOSE



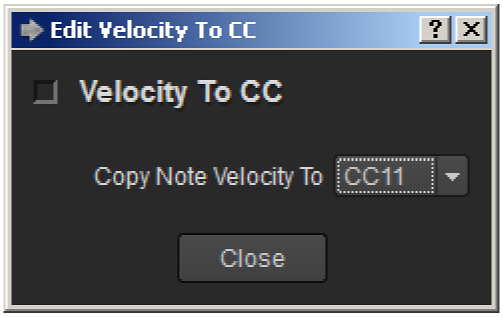
The **Transpose** transform simply transposes all notes a certain number of semitones. Enter a positive number to transpose up, for instance enter 12 to transpose up one octave. Enter a negative number to transpose down, for instance enter -3 to transpose down a minor third.

VELOCITY MAP



The **Velocity Map** transform is used to modify the velocities of notes. It utilizes a **Value Map Editor** with the incoming note velocity on the x-axis and the outgoing note velocity on the y-axis. The **Value Map Editor** is described in more detail later on.

VELOCITY TO CC

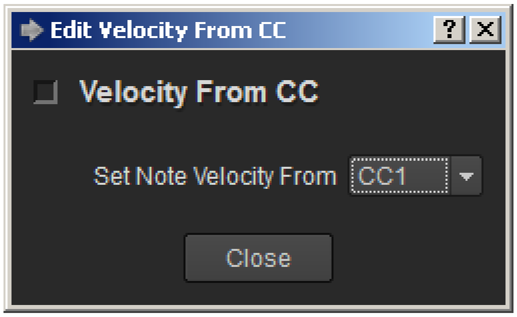


□ Velocity To CC

Copy Note Velocity To CC11

The **Velocity To CC** transform is used to inject an additional continuous controller (CC) message for every note received, with its value being the same as the note velocity. In the example above, a CC11 message will be injected in the MIDI data stream for each received note. If the note's velocity is 88, the CC11 message created will have a value of 88.

VELOCITY FROM CC



□ Velocity From CC

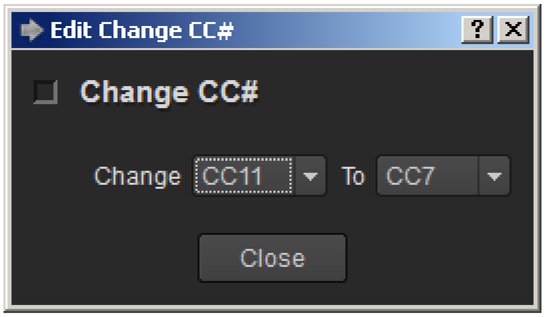
Set Note Velocity From CC1

The **Velocity From CC** transform is used to control note velocities from a continuous controller (CC). Received notes will have their velocities set to the most recently received value of the CC specified.

CONTROL TRANSFORMS

Control transforms either operate on or are derived from continuous controller (CC) messages. The various control transforms are described below.

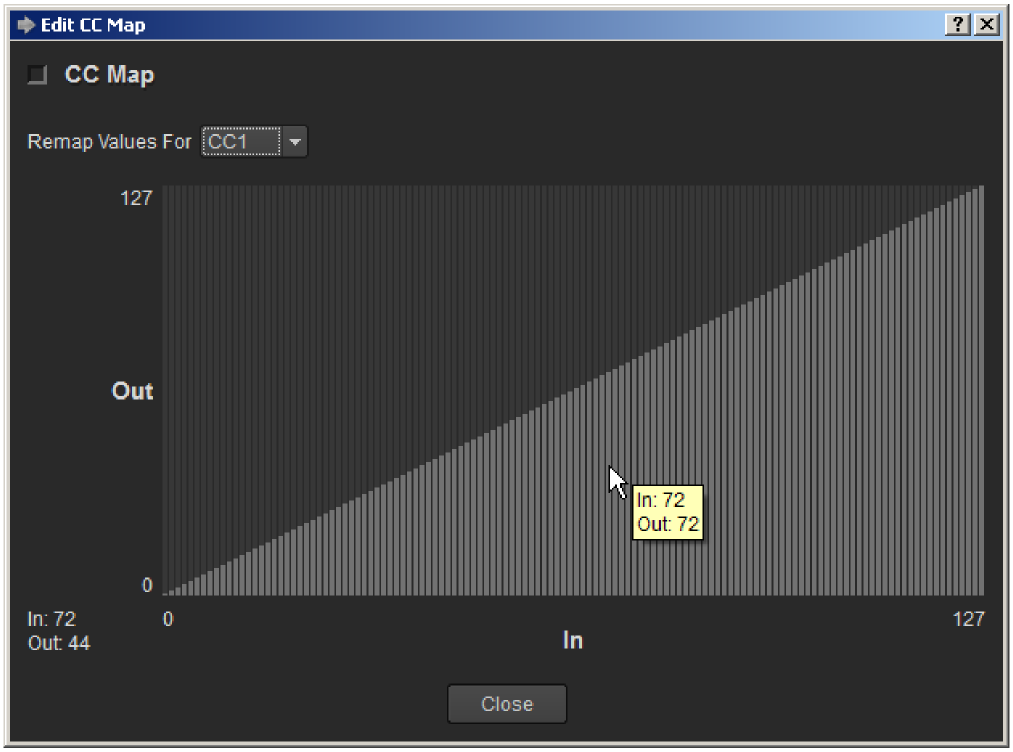
CHANGE CC#



Change CC11 To CC7

The **Change CC#** transform is used to change the continuous controller number ("CC#") of a particular type of continuous controller. For instance, the example above changes CC11 (expression) values to CC7 (volume).

CC MAP

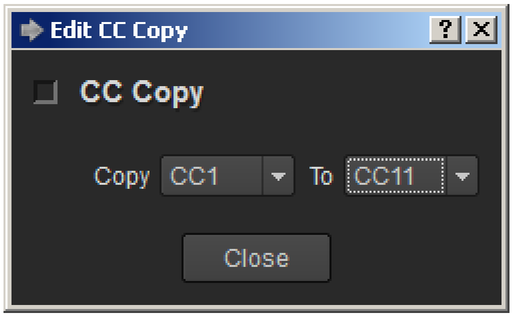


Remap Values For CC1

In: 72
Out: 44

The **CC Map** transform is used to remap continuous controller (CC) values. It utilizes a **Value Map Editor** with the incoming CC value on the x-axis and the outgoing value on the y-axis.

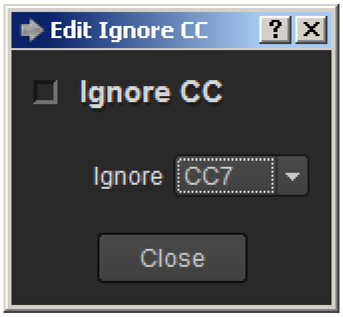
CC COPY



Copy CC1 To CC11

The **CC Copy** transform is used to inject additional continuous controller (CC) messages when the specified continuous controller messages are received. For the example shown above, whenever a CC1 message is received, an additional CC11 message will be injected into the MIDI data stream, with the same value as the incoming CC1 message.

IGNORE CC



Ignore CC7

The **Ignore CC** transform ignores the continuous controller (CC) messages from the controller specified. Messages from the specified controller will not be passed to other transforms, nor will they be sent to the MIDI output port.

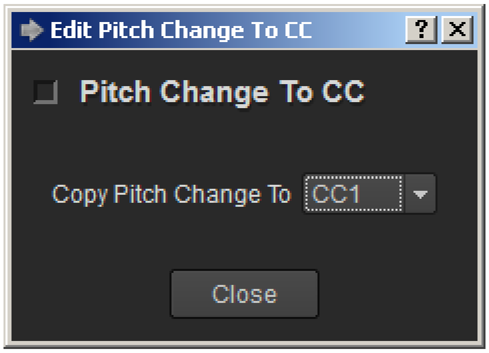
PITCH CHANGE TRANSFORMS

Pitch change transforms operate on or are derived from pitch change (also called “pitch bend” and “pitch wheel”) messages.

INVERT PITCH CHANGE

The **Invert Pitch Change** transform simply inverts the value any pitch change messages. It makes a bend up go down and a bend down go up. *Note: no editor is needed.*

PITCH CHANGE TO CC



□ Pitch Change To CC

Copy Pitch Change To CC1

The **Pitch Change To CC** transform creates new CC messages with the control number specified, according to the value of the incoming pitch change value. In the example above, if the pitch wheel is all the way down, a CC1 value of 0 will be created. If the pitch wheel is all the way up, a CC1 value of 127 will be created.

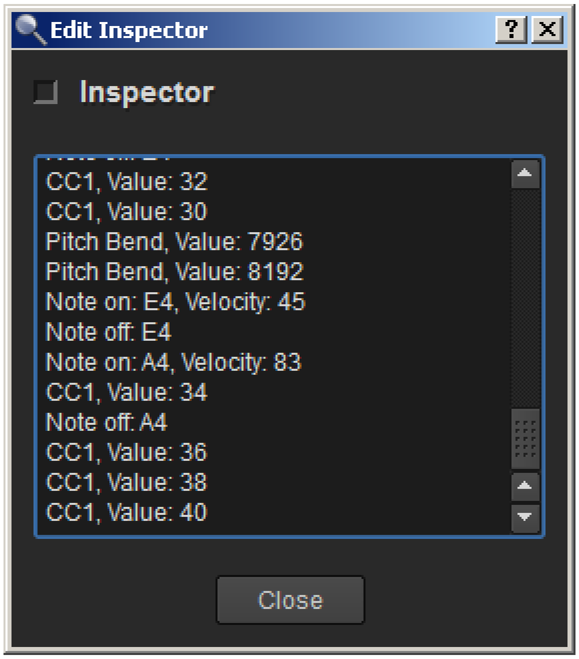
IGNORE PITCH CHANGE

The **Ignore Pitch Change** transform is similar to the **Ignore CC** transform, except pitch change messages are ignored instead of a particular continuous controller. Pitch change messages will not be passed to other transforms, nor will they be sent to the MIDI output port. *Note: no editor is needed.*

UTILITY TRANSFORMS

Utility transforms are transforms that don't belong in any of the other categories, as they might operate on all incoming MIDI messages or perform some strange operation. Currently the only utility transform is the **Inspector**, described below.

INSPECTOR



```
CC1, Value: 32
CC1, Value: 30
Pitch Bend, Value: 7926
Pitch Bend, Value: 8192
Note on: E4, Velocity: 45
Note off: E4
Note on: A4, Velocity: 83
CC1, Value: 34
Note off: A4
CC1, Value: 36
CC1, Value: 38
CC1, Value: 40
```

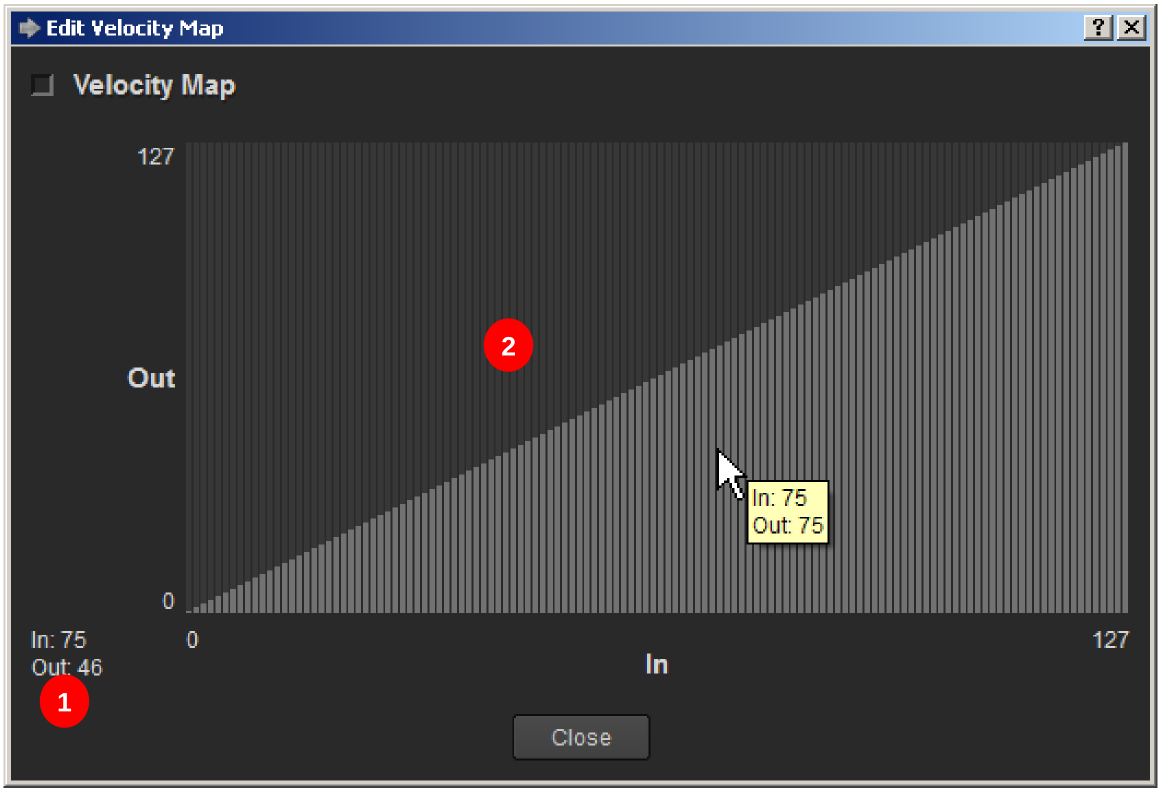
The **Inspector** really isn't a transform as much as it is a debugging tool. It doesn't modify any MIDI data. What it does is display the MIDI messages it receives. This is useful in order to see what MIDI messages are being received by a patch or what upstream transforms are doing.

TRANSFORM CHAINING

As you've probably noticed, each transform performs a very specific, simple task. More involved, powerful transformations can be achieved by using a series of transforms within a patch, as the output of each transform is available to the transforms that follow it (messages moving from left to right). Also, since each patch within an output can have its own set of unique transforms, the MIDI output from each patch can be completely different from the MIDI signals received by the parent output, even though all patches receive the exact same input! This can make for some very interesting sonic possibilities.

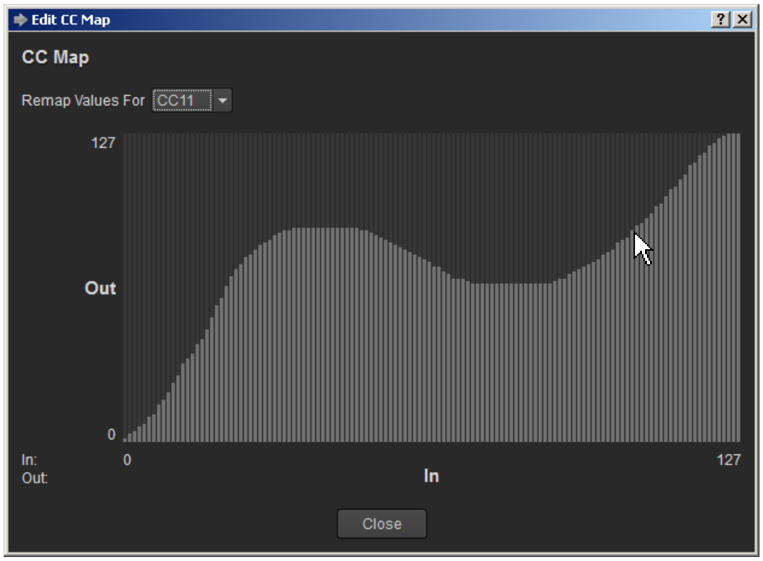
VALUE MAP EDITING

Value Map Editors are used in many of the transform editors. The value map is a simple mapping of input value to output value. Input values (typically 0 to 127) are shown along the x-axis and output values (also typically 0-127) are shown along the y-axis. For an example, let's take a look at the **Velocity Map** editor:



1. Current position display. As the mouse cursor moves over the editing area, the input/output at the cursor position is shown here.
2. Editing area. Simply press and move the cursor to draw curves, use the middle mouse button (or Alt+press) to draw lines, or use the right-click context menu for more options.

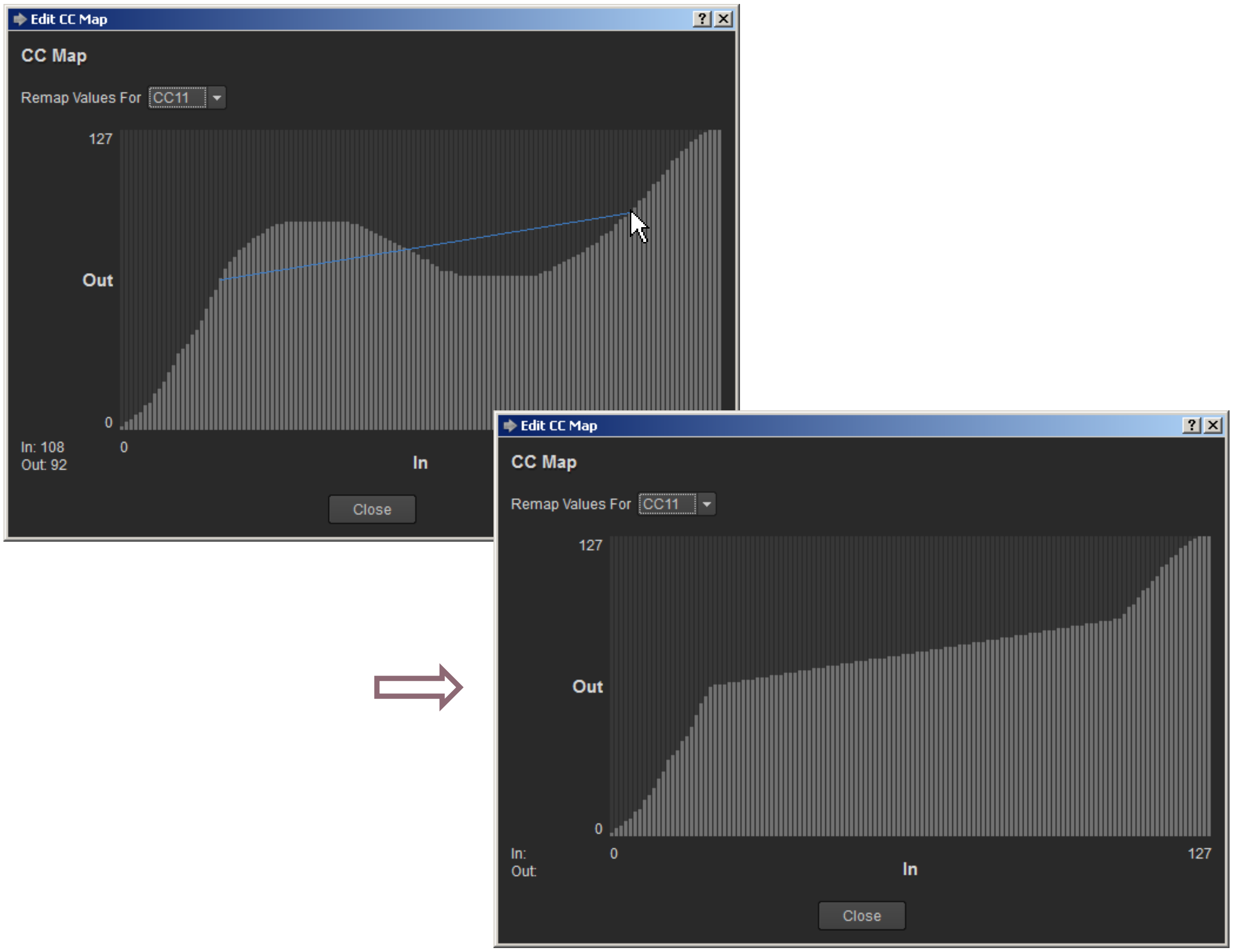
Values can be edited free form using the mouse by simply pressing and dragging the mouse cursor around:



Remap Values For CC11

In:
Out:

The middle mouse button (or Alt+press) and drag can be used to make linear value ramps:



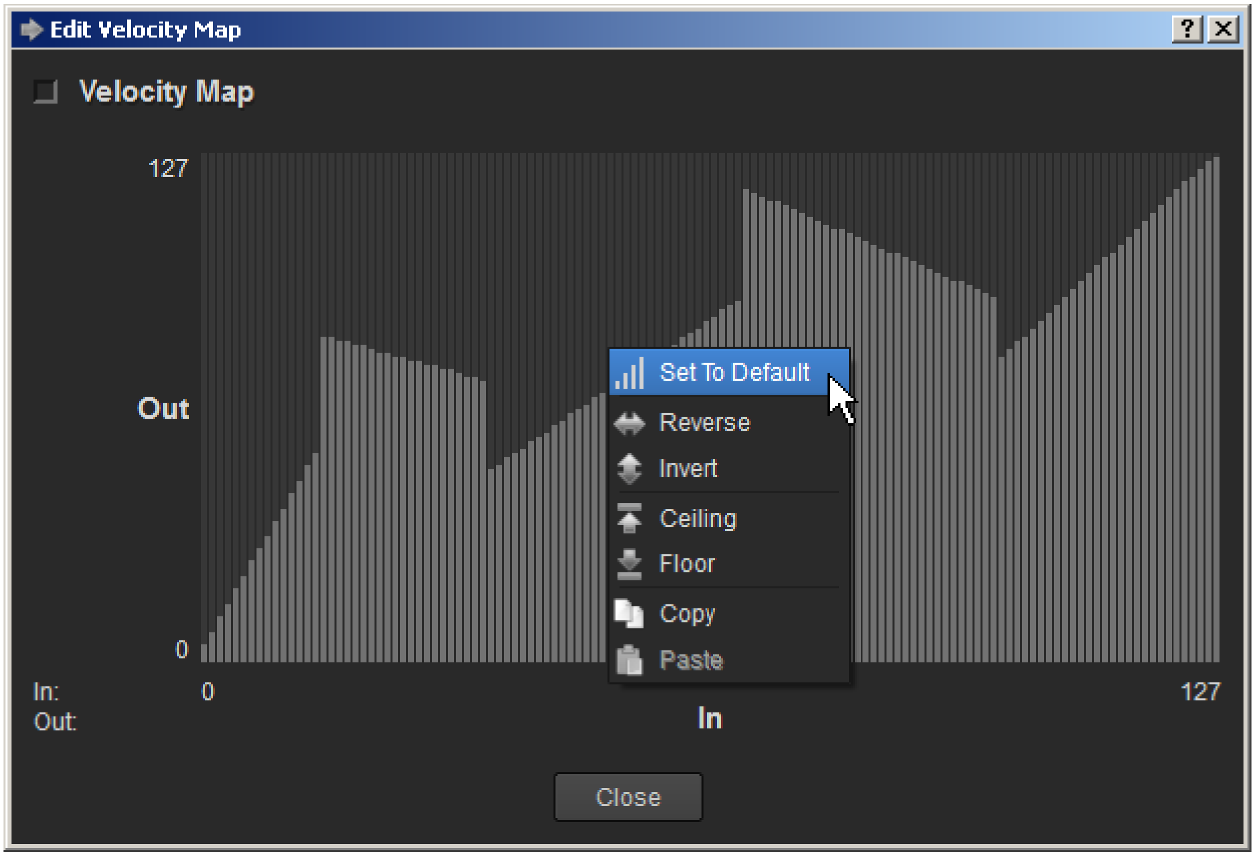
Remap Values For CC11

In: 108
Out: 92

Remap Values For CC11

In:
Out:

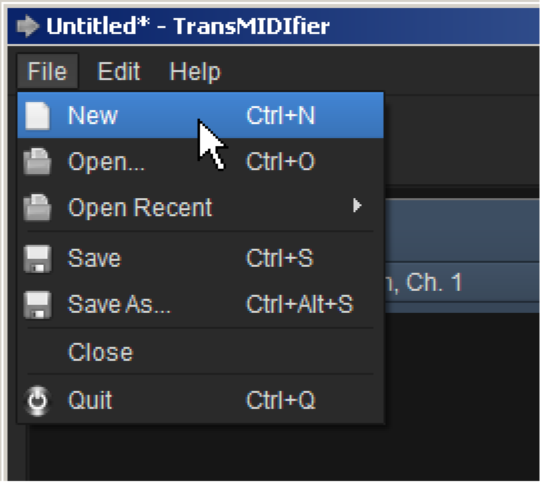
The Value Map Editor also has a right-click context menu that provides some handy operations:



- **Set To Default**: sets output equal to input (effectively a bypass of the transform).
- **Reverse**: reverses the current value mappings left-to-right, so the output value for input 0 becomes the output value for input 127 and vice versa, all the way across.
- **Invert**: inverts the current value mappings by changing every value to be 127 - (that value).
- **Ceiling**: set the output value for all input values to be the maximum value (typically 127).
- **Floor**: set the output value for all input values to be the minimum value (typically 0).
- **Copy**: copy the value mapping to the clipboard, so it can be pasted into another value map editor.
- **Paste:** paste whatever value mapping is currently present in the clipboard, overwriting the current values.

SAVING AND LOADING

Once you’ve got a useful set of inputs, outputs, patches, and transforms in your setup, you can save the setup to disk so as to be recalled later. The save/load options are inside the **File** menu:



- **New**: create a new, empty setup.
- **Open…**: browse for an existing setup file to open.
- **Open Recent**: gives a menu of recent setup files from which to choose.
- **Save**: save the current setup. If the current setup hasn’t been saved before you’ll be prompted for a save location.
- **Save As…**: choose a new location at which to save the setup.
- **Close**: close the current setup.
- **Quit**: exit the application.

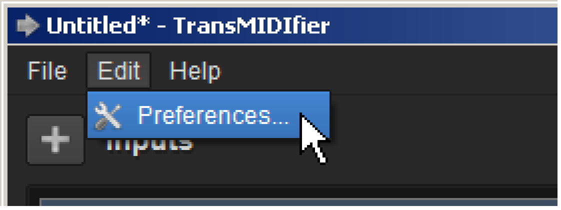
Note: the menus are slightly different on Mac OS X. Take a moment to look around within the available menus – you should be able to everything mentioned here easily enough.

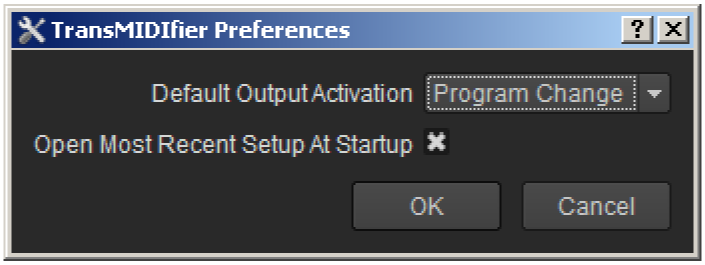
OBJECT REORDERING

Whenever a new object is added to the setup, be it an input, output, patch, or transform, it is added “at the end” of the current set of that type of object. However, objects can easily be reordered by simply pressing and dragging to the desired position.

PREFERENCES

The preferences dialog is launched from the **Edit** menu:




TransMIDIfier Preferences

Default Output Activation: Program Change

Open Most Recent Setup At Startup ☒

- **Default Output Activation**: this sets the default activation method for new MIDI outputs to be either Keyswitch or Program Change.
- **Open Most Recent Setup At Startup**: when checked, the most recent setup file will be loaded when the application is launched.