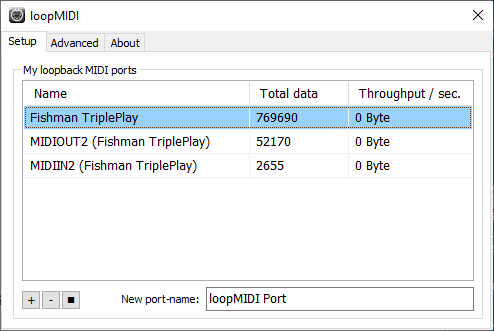
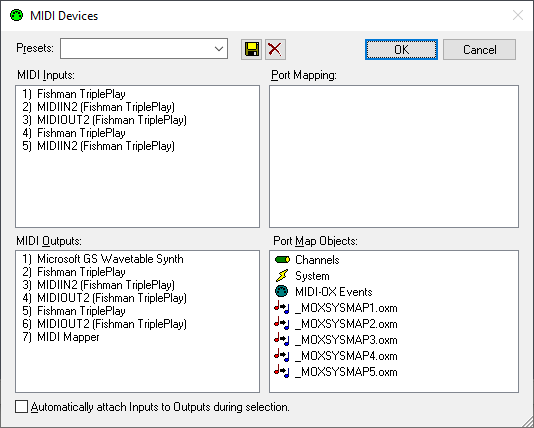
2020-06-14 OK, so I think I finally figured out how to spoof the FTP Editor using MidiOX and loopMidi.

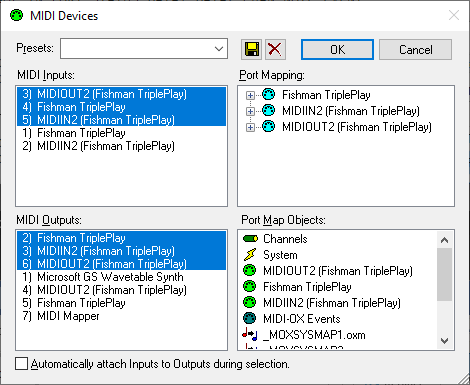
1. **loopMidi Setup**
2. “FTP” stands for “Fishman TriplePlay” in endpoint names
3. Create loopMidi devices named “MIDIIN2 (FTP)”, MIDIOUT2 (FTP), and “FTP”



**(2) select MidiOx ports.**

In midiOx, the loopMidi ports will be listed first



So, in the above window, Inputs 1,2, and 3 are loop Midi ports, but 4 and 5 are the “real” ports, and Outputs 2,3, and 4 are “fake” and 5 and 6 are real.

1. Select the following inputs: 3, 4, and 5 (real inputs, fake output)

3 = the ‘fake’ MIDIOUT2

4 = the ‘real’ FTP input

5 = the ‘real’ MIDIIN2 input (data FROM the FTP editor)

1. Select the following outputs: 2,3, and 6

2 = the ‘fake’ FTP (output to FTP editor)

3 = the ‘fake’ MIDIIN2 (output to FTP editor)

6 = the ‘real’ MIDIOUT2

When viewed later in MidiOx, just remember that higher numbers are ‘real’ and lower numbers are ‘fake’

* + 1. Spoofing the FTP editor, continued …

**IMPORTANT NOTES:**

**IT IS CRUCIAL THAT THE ‘FAKE’ PORTS COME BEFORE THE ‘REAL’ ONES IN THE ABOVE LISTS.**

YOU MAY HAVE TO UNINSTALL the FTP “Sound Video or Game Controller” device from windows, make sure the dongle is not in the machine, set the loopMidi ports, then reboot, START loopMidi again, and re-insert the dongle to get them in the correct order.

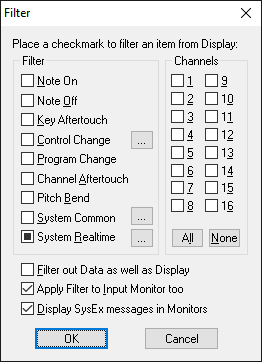
**IF THEY ARE NOT IN THE CORRECT ORDER, the FTP editor will mysteriously just not work**, as it appears to take the FIRST ports it finds BY NAME in the list, and if it find the ‘real’ thing, but cannot open it (because we have taken control of it), the stupid effing FTP editor will not try another port, will not give any message … it will just not work.

**SO MAKE SURE THE PORTS ARE IN ROUGHLY THE ORDER ABOVE … WITH THE LOOPMIDI PORTS LISTED BEFORE THE real FTP ports.**

**ALSO NOTE:**

****

**THAT BLUE SYSEX BUTTON MUST BE ON!!! It is a full-on filter. If you just don’t wanna see sysex messages, that’s the last checkbox in the “Options-Midi Filter” Dialog:**



2020-06-14 Spoofing the FTP editor, continued …

**(3) Setup the port routings in midiOx,** connecting things with like names, and connecting the midiOx “event port” to the “real” MIDIOUT2 …



At this time, you should be able to run the FTP editor, it should come up, and you should get a bunch of messages in midiOx.

**IF YOU QUIT MIDI-OX you have to re-do the port selection and routing.** Even though it looks like it remembers things, it only associates the ports by name when it restarts, and so picks the wrong ones!

2020-06-14 Spoofing the FTP editor, continued …

**(4) GENERAL NOTES**

The FTP Editor program comes up with something like the above, and the midiOx input event monitor window will contain a slew of stuff: Note that I have realtime messages (FE = active sense) filtered out of both windows.



It is worth noting at this point that the notion of MidiOx’s “input” and “output” monitors is not useful, as ALL messages will be displayed in BOTH windows. But what IS useful is that the messages are listed according to the port that sent them, so in the above window you can tell what the Editor is sending and the Controller is responding with:

TIMESTAMP IN PORT STATUS DATA1 DATA2 CHAN NOTE EVENT

0000EB46 3 -- F0 Buffer: 6 Bytes System Exclusive

SYSX: F0 7E 00 06 01 F7

0000EB48 5 -- F0 Buffer: 17 Bytes System Exclusive

SYSX: F0 7E 00 06 02 00 01 6E 00 01 00 02 01 55 01 00 F7

0000EB48 5 -- F0 Buffer: 17 Bytes System Exclusive

SYSX: F0 7E 00 06 02 00 01 6E 00 01 00 02 01 55 01 00 F7

0000ED3A 3 -- B7 1F 04 8 --- Control Change

0000ED3A 3 -- B7 3F 02 8 --- Control Change

...

The first message is from “IN” #3, which is the editor sending a message to MIDIOUT2, in this case a sysex “identity request” F0 7E 00 06 01 F7.

The controller replies from “IN” port #5 (MIDIIN2?) with the identity message F0 7E 00 06 02 00 01 6E 00 01 00 02 01 55 01 00 F7, which I have sort of documented elsewhere.

2020-06-14 Spoofing the FTP editor, continued …

**(5) Setting the FTP String Sensitivity**

It appears as if many FTP control communications are set up using a pair of NRPNs where

B7 1F xx command (or reply) xx

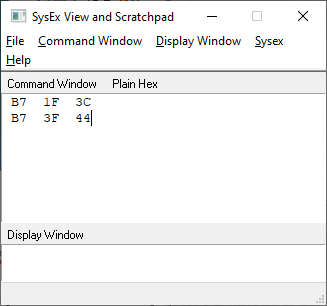
B7 3F yy command (or reply) value yy

One such pair of commands are the “set” and “get” string sensitivity commands, 0x3C and 0x42

xx == 0x3C == command “set string sensitivity”

xx == 0x42 == command “get string sensitivity”

**SET SENSITIVITY**



You can set the sensitivity of the strings by sending the following commands to the FTP.

**B7 1F 3C command “set string sensitivity”**

**B7 3F xy value**

Where **x** is the string number (from 0..5) and **y** is the sensitivity (from 0..15). This is best done in midiOxusing the **“View-Sysex” window**, which can also send arbitrary bytes ….

The window to the right, which sends 3F 44, sends the command to set string 5 to sensitivity 5 (remember it’s all it’s all zero based).

**GET SENSITIVITY**

You get the string sensitivity by sending the following where **xx** is the zero based string number, and the controller will reply with a pair of it’s own NRPNs containing yy the value currently assigned to the string:

**Send: B7 1F 42**

**B7 3F xx xx =** string number

**Recv: B7 1F 42**

**B7 3F yy** yy = current sensitivity value