MIDI UWP Router

Configurable router for MIDI, using UWP

Preface

Windows has for MIDI two systems: Microsoft Multimedia Extension (MME) and exposure of the devices using Unified Windows Platform (UWP/WinRT). MME exists since Windows 95, UWP started somewhere around 2016. The difference with regards to possibilities is small. Main difference is the naming schema and UWP also supporting MDI using Bluetooth Low Energy devices (BLE). MME does *not* support BLE.

Most of the existing DAWs and Live Performance Plugin hosts do not support UWP and therefore also do not support BLE. This is where this router comes in: It opens up the possibility to receive MIDI-messages from a BLE device and send them to a 'normal' MIDI-device. If the latter is a virtual MIDI cable, like loopMIDI (Tobias Errichsen), then the DAW or Live Performance Plugin Host can listen for incoming messages using the output side of that virtual MIDI cable.

Naming

As mentioned above, the naming schemas of MME and UWP differ. MME tends to use a symbolic name, UWP passes through the name at the device level (also visible in devicemanager). Despite of UWP messing up some names, for example those of the outputs of loopMIDI, UWP opens up more possibilities for recognising devices, by also exposing the device IDs.

CookedNames

Some devices are assigned the non-descriptive name MIDI (for example output devices of loopMIDI). To compensate for this MIDIUWPRouter cooks a name by looking up the associated input device (if any) and using its name. These are referred to by the term 'CookedName'. This only applies to **output** devices of which real names are `'MIDI'` or derivates like `'2 - MIDI'`. For all other output devices (and input devices as well), CookedName is assigned the RealName.

Configuration

The MIDIUWPRouter needs a configuration file to tell it which input should be forwarded to a certain output. This config file has the following layout:

```
[MIDIIN=<alias-name>]
Filter=Name:<some name>
Filter=RealName:<some real name>
Filter=RegEx:<regular expression of a name>
Filter=RealRegEx:<regular expression of a real name>
Filter=ID:<full ID>
<multiple filters allowed, first match is applied>

[MIDIIN=<alias-name>]
<etc>

[MIDIOUT=<alias-name>]
Filter=Name:<some name>
Filter=RealName:<some real name>
Filter=RegEx:<regular expression of a name>
Filter=RealRegEx:<regular expression of a real name>
Filter=ID:<full ID>
<multiple filters allowed, first match is applied>

[MIDIOUT=<alias-name>]
<etc>

[ROUTE=<route-name>]
SOURCE=<alias-name>
DESTINATION=<alias-name>

#next route is commented out by adding a hash-sign **after** the section header
[ROUTE=<route-name>]#
SOURCE=<alias-name>
DESTINATION=<alias-name>

[ROUTE=<route name>]
<etc>
```

Remarks

- A router configuration file can be created using notepad or other simple text editors. Wordpad, Write and Word are *not* suitable for this task.

- There must be no space before and after the "=" signes. A space before the colon (":") is also not allowed when defining filter types (`FILTER=ID:arg`).
- Alias-names and route-names must be at least 3 characters long.
- MidiIn-Alias-names, MidiOut-Alias-names and route-names must be unique.
- Alias-names are internally converted to uppercase and thus case-insensitive.
- Alias-names cannot contain a colon (:).
- Duplicate routes (route-a-source == route-b-source and route-a-destination == route-b-destination) are not allowed.
- Keywords like `MIDIIN=` , `MIDIOUT=` , `ROUTE=` , 'FILTER= `,` DESTINATION= ID: `,` REGEX:`, etc. are not case-sensitive. However, the arguments for the filters are case sensitive.
- **Warning** Be careful when trying to comment out a section by putting a character in front of a section header. This will result in applying the next lines to the current section. Better is to add a character at the end of the section header like `[MIDIIN=something]#` .

Explanation of `[MIDIIN]` and `[MIDIOUT]` definitions

- `[MIDIIN=<alias-name>]` and `[MIDIOUT=<alias-name>]`
  This header defines a source alias or a destination alias. The `alias-name` is to be used in the router definitions. After this header one or more filter rules must follow.

  TIP Alias definitions are processed in the order they are defined. A device can only be associated with one alias.
  TIP Devices are sorted by RealName before parsing of the configuration file starts.

  - `FILTER=<filter type>:<argument>` A filter describes how a device must be associated with the alias. There can be more than one filter.

    TIP Filters are processed in the order they are defined and the first match wins.
    TIP Device IDs, CookedNames and RealNames can be retrieved by starting the router without any config. This will dump all available MIDI devices: Their CookedNames, RealNames and IDs

    There are five possible filter types:

    - `FILTER=ID:<filter argument>` The `filter argument` is a full device ID.

- `FILTER=Name:<filter argument>` The `filter argument` is a full CookedName of the device.
- `FILTER=RegEx:<filter argument>` The `filter argument` is a regex. The CookedNames of the devices are matched against this regex.
- `FILTER=RealName:<filter argument>` The `filter argument` is a full RealName of the device.
- `FILTER=RealRegEx:<filter argument>` The `filter argument` is a regex. The RealNames of the devices are matched against this regex.

Explanation of `[ROUTE]` definitions

- `[ROUTE=<route name>]`
  Defines a route from an MIDIIN-alias to a MIDIOUT-alias. The `<route name>` is only used to indentify routes while parsing the config file. A route definition needs two extra lines:

  TIP An alias can take part in more than one route. When there is no device associated with a source alias or with a destination alias, the route will not be instantiated, but skipped. This is not an error. It is always possible that a MIDI device is not connected at one particular time

  - `SOURCE=<alias name>`
    Defines the alias of the device acting as a source of MIDI messages
  - `DESTINATION=<alias name>`
    Defines the alias of the device that will receive the MIDI messages

Starting MIDIUWPRouter

To get IDs, RealNames and CookedNames of devices, just start MIDIUWPRouter without any arguments. When the path of a config file is added to the command, that configuration will be applied and the router will start routing.

Examples:

- `MIDIUWPRouter.exe`
- `MIDIUWPRouter.exe "d:\music\routerconfig.txt"`