Virtual Devices & Event Injection

This project supports “virtual” (software) MIDI devices that participate in the same device lists and event pipeline as hardware/network devices.

This is used by optional integrations (e.g., Maestro / MPTK), but it’s also useful for:

- automated tests
- simulation tools
- bridging other input sources into `MidiManager`
- creating a software “loopback” device

Implementation location:

- `Assets/MIDI/Scripts/MidiManager.VirtualDevices.cs`

What is a virtual device?

A **virtual device** is identified by a `deviceId` string (you choose the convention). After registration, it can show up in:

- `MidiManager.Instance.DeviceIdSet`
- `MidiManager.Instance.InputDeviceIdSet`
- `MidiManager.Instance.OutputDeviceIdSet`

...and it triggers the same attach/detach callbacks your app would receive from real devices.

Register / unregister

Register a virtual device as input and/or output:

- `RegisterVirtualMidiDevice(deviceId, input: true/false, output: true/false)`
- `UnregisterVirtualMidiDevice(deviceId)`

Recommended:

- Use a clear prefix like `virtual:...` or `mptk:...`.
- Use `group = 0` unless you intentionally model multiple groups.

Injecting events (“pretend this device sent it”)

Virtual devices can inject MIDI messages into the manager’s internal pipeline:

- Note On/Off
- CC, Program Change
- Aftertouch, Pitch Wheel
- SysEx
- System common / realtime (where applicable)

Important:

- **Injection does not send to hardware/backends.**
- Injection is for software-generated input, as if it arrived from `deviceId`.

This is how “MPTK as a virtual input source” works.

See also:

- Maestro / MPTK Integration