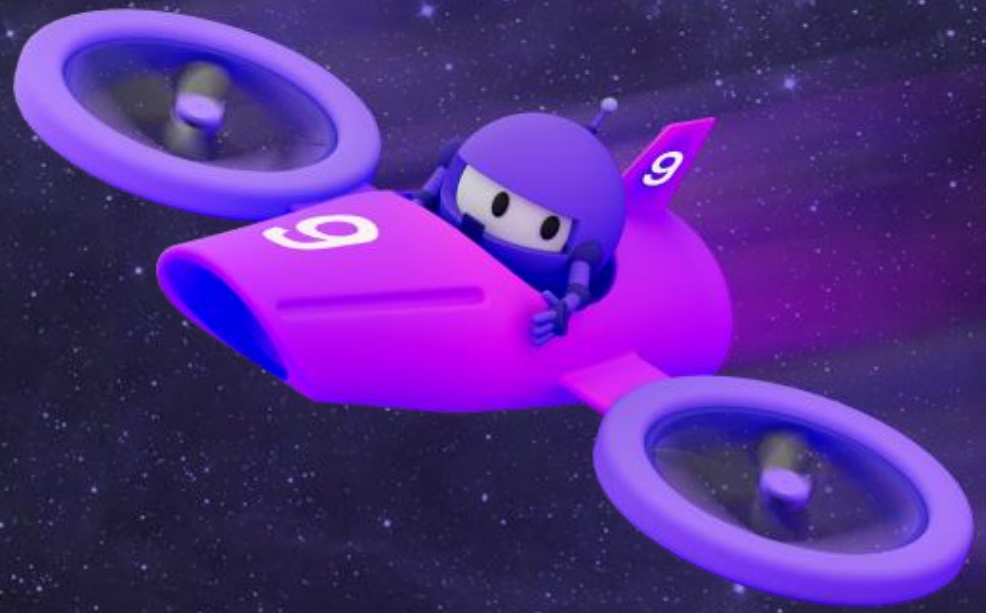
Create great MIDI 2.0 apps using Windows MIDI Services and C#



Pete Brown

Principal Engineer – Windows
Windows MIDI Services Project lead
MIDI Association Executive Board Chair

Pete Brown

Agenda

00

What is MIDI?

What is MIDI 2.0

01

Windows MIDI Services

Using Windows MIDI Services from C# and dotnet

02

Best Practices

Where to go to learn more


00
What is MIDI?
What is MIDI 2.0?

MIDI (v1.0 1983)

Musical Instrument Digital Interface original byte format

Data exchange protocol for musical instruments, stage lighting, laser shows, and more

Transfers control data, not audio "Play C# 3rd Octave at 50% velocity on channel 2" or "Set the filter frequency to 25% on channel 16"

Example: Play C# 3rd Octave at 50% velocity on channel 2

	Status: Opcode (high bit always 1) Remaining [0..7]	Status: Channel (indexes [0…15])	Data 1: Note Number (indexes [0…127])	Data 2: Velocity (indexes [0…127])
Decimal	9 (1 + high bit)	1 (index: 1 == ch 2)	61 (C# third octave)	64 (rounded)
Binary	1001	0001	0011 1101	0100 0000

MIDI-CI (v1.0 2018, v1.2 2023)

MIDI Capability Inquiry

MIDI-CI is a bridge between MIDI 1.0 and MIDI 2.0
Considered part of the MIDI 2.0 Family

Runs over MIDI 1.0 and MIDI 2.0 transports
Implemented as standardized System Exclusive (SysEx) messages
Requires bidirectional communication (USB, DIN pairs, BLE, etc.)

Main features are Profiles "I'm a piano" and Property Exchange "My current patch is A-43 : Burning Grand"

MIDI 2.0 UMP (v1.1 2023)

Universal MIDI Packet format

Packet-based, not byte stream based
UMP Words are 32 bits. Packets are 1-4 words long, set by the message type.

Same Channel index rules apply (0-15 == 1-16).
Group number follows same rules.

Example: Play C# 3rd Octave at 50% velocity on channel 2 on group 5

	Message Type	Group	Opcode	Channel	Note Number	Attribute Type
Word 0	0100 (4)	0100 (4)	1001 (9)	0001 (1)	0011 1101 (61)	00000000 (0)

	Velocity (16 bits)	Attribute (16 bits)
Word 1	1000 0000 0000 0000 (32768)	0000 0000 0000 0000 (0)

MIDI 2.0 Devices today

Yamaha Montage M

Fully supports the Universal MIDI Packet format to send and receive high-resolution MIDI 2.0 data

Roland A-88

Fully supports the Universal MIDI Packet format to send and receive high-resolution MIDI 2.0 data

NI Komplete Kontrol S

Native Instruments Komplete Kontrol S Uses the new Universal MIDI Packet format to send and receive MIDI 1.0 data

Korg Keystage

Supports the MIDI-CI aspect of MIDI 2.0, which enables bidirectional communication between the controller and the DAW

Apps

Steinberg Cubase and Bremmers MultitrackStudio both have prototypes using Windows MIDI Services

Apple's Logic Pro supports MIDI 2.0 UMP, as do several DAWs on macOS all via Core MIDI.

More

More devices and apps coming! (They're mostly waiting for us to get Windows MIDI Services in-box ☺

01

Windows MIDI Services

Windows MIDI Services

Unified MIDI 1.0/2.0 API

Supports MIDI 1.0 through the Universal MIDI Packet. Translation, when needed, happens in the service

New USB Driver

Brand-new MIDI 1.0/MIDI 2.0 driver with faster data transfer and new features

Multi-client

All devices are now multi-client, enabling multiple applications to use the same MIDI device at the same time

Virtual / Loopback Devices

Built-in Virtual Devices and simple loopback support, without the need of third-party drivers.

Timestamped

Full support for timestamps and outgoing message scheduling

Backwards Compatible

WinMM API re-plumbed to talk to the new service and API

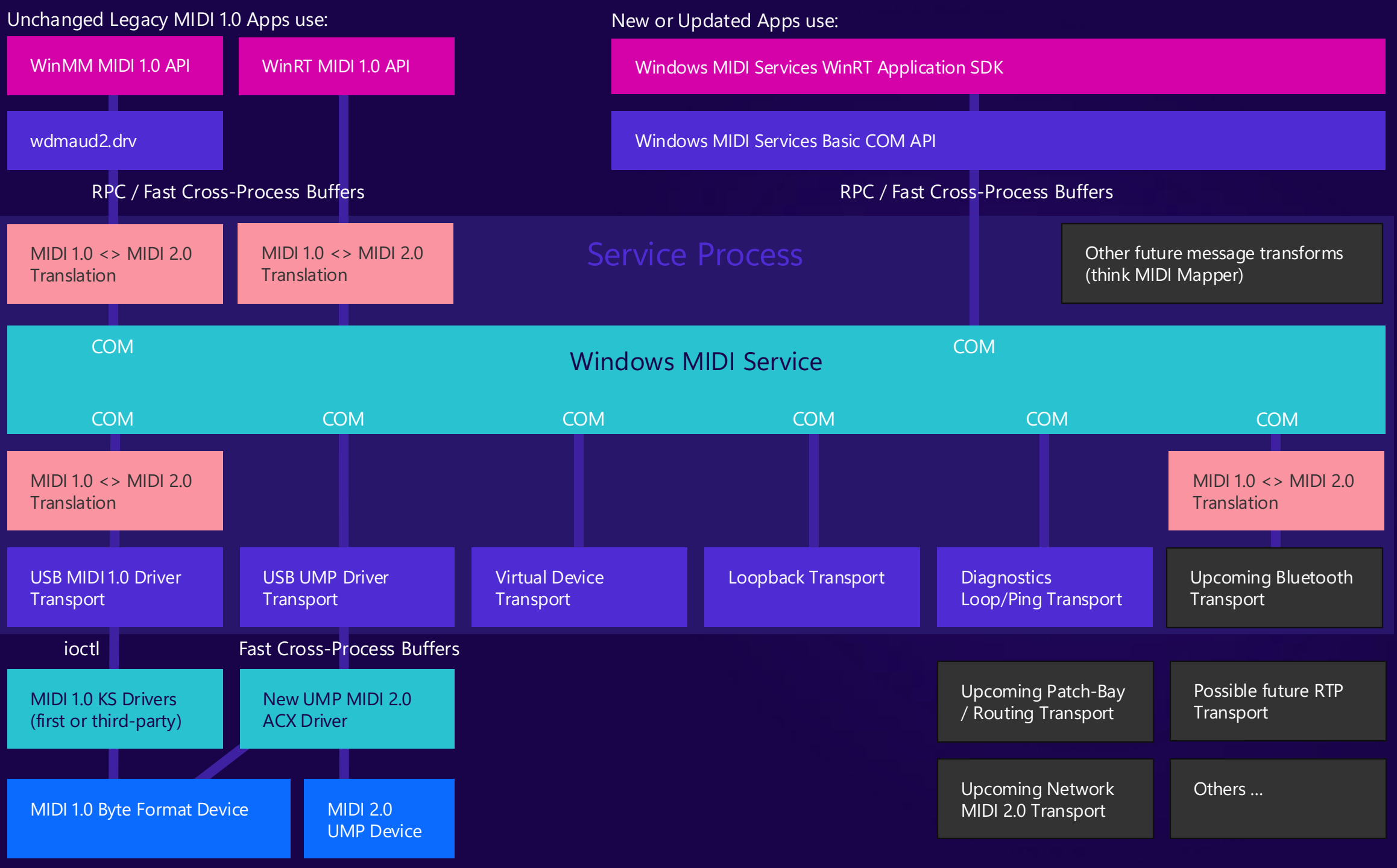
Built for Expansion

New plugin approach for transports like Network MIDI 2.0, Bluetooth, and more.

Network MIDI 2.0 will be shown at NAMM in January 2025

Open Source

Created iteratively with partners on Github and ingested into Windows. Driver, service, API, SDK, tools are all MIT-licensed OSS.


Unchanged Legacy MIDI 1.0 Apps use:
WinMM MIDI 1.0 API
WinRT MIDI 1.0 API
wdmaud2.drv
RPC / Fast Cross-Process Buffers
New or Updated Apps use:
Windows MIDI Services WinRT Application SDK
Windows MIDI Services Basic COM API
RPC / Fast Cross-Process Buffers
MIDI 1.0 <> MIDI 2.0 Translation
MIDI 1.0 <> MIDI 2.0 Translation
Service Process
Other future message transforms (think MIDI Mapper)
COM
Windows MIDI Service
COM
COM
COM
COM
COM
COM
COM
MIDI 1.0 <> MIDI 2.0 Translation
MIDI 1.0 <> MIDI 2.0 Translation
USB MIDI 1.0 Driver Transport
USB UMP Driver Transport
Virtual Device Transport
Loopback Transport
Diagnostics Loop/Ping Transport
Upcoming Bluetooth Transport
ioctl
Fast Cross-Process Buffers
MIDI 1.0 KS Drivers (first or third-party)
New UMP MIDI 2.0 ACX Driver
Upcoming Patch-Bay / Routing Transport
Possible future RTP Transport
Upcoming Network MIDI 2.0 Transport
Others ...
MIDI 1.0 Byte Format Device
MIDI 2.0 UMP Device

Demo

Windows MIDI Services App SDK

- How to use Windows MIDI Services from C#

```
using Microsoft.Windows.Devices.Midi2;
using Microsoft.Windows.Devices.Midi2.Diagnostics;
using Microsoft.Windows.Devices.Midi2.Initialization;

MidiServicesInitializer.EnsureServiceAvailable();

using var session = MidiSession.Create("Dotnet Conf 2024") ;

var inputConnection = session.CreateEndpointConnection(MidiDiagr

if (inputConnection ≠ null)
{
    inputConnection.MessageReceived += (sender, e) ⇒
    {
        var message = e.GetMessagePacket();

        Console.WriteLine($"At {message.Timestamp}, Received: {m
```

02

Best Practices and How to Learn More

Best Practices

Use the WinRT App SDK

The WinRT SDK includes features to ensure your apps can make the most of MIDI 2.0. The underlying COM API includes only what is needed to talk to the service.

Use .NET 8 or .NET 9

The WinRT SDK supports C++, and .NET 6 or better, but is best used with .NET 8 or .NET 9 when using .NET.

One connection per Endpoint

Each connection uses resources. If possible, keep the number of connections to a single endpoint to just one per application/session.

Use the Endpoint Watcher

Use the `MidiEndpointDeviceWatcher` to get notifications of property updates and device connect/disconnect

Schedule fewer messages

Try to keep the outgoing message scheduler to < 500 messages per endpoint.

Send immediate messages

For sending messages immediately, use a timestamp of 0 (also a constant on the `MidiClock` type)

Learn more at https://aka.ms/midi

Documentation
Repo links
Discord links
Spec links

Thank you

My original astrophototography: https://www.flickr.com/photos/psychlist1972/albums/