Maestro / MPTK Integration (Virtual Devices + Adapters)

This page documents the optional **Maestro / MPTK (MidiPlayerTK)** integration layer.

It provides two high-level workflows:

1. **MPTK as a virtual MIDI output device (sink)**
   Route `MidiManager` events into an MPTK synth (realtime playback).
2. **MPTK as a virtual MIDI input device (source)**
   Take MPTK callbacks (from a `MidiFilePlayer` or `MidiStreamPlayer`) and **inject** equivalent MIDI events into `MidiManager` as if they came from a device.

> Terminology note: in this plugin, "virtual device" means a software endpoint that participates in `MidiManager`'s device lists and event pipeline without involving platform MIDI backends.

Enable / Disable (compile-time)

The integration is gated behind a scripting define:

- `FEATURE_USE_MPTK`

When the define is **not** present (or MPTK is not installed), the integration scripts compile to small stubs so the project still builds.

How to enable

In Unity:

- **Project Settings → Player → Other Settings → Scripting Define Symbols**
- Add: `FEATURE_USE_MPTK`

What's included

Virtual device registration + input injection

`MidiManager` has helper APIs for software endpoints:

- Register/unregister a virtual device ID (as input, output, or both)
- Inject MIDI events directly into the internal pipeline ("pretend this device sent a Note On", etc.)

Use these when you want to simulate devices, bridge other systems, or unit-test handlers.

MPTK virtual MIDI device (sink)

`MptkMidiDevice` implements MIDI event handler interfaces and forwards events to an internal MPTK `MidiStreamPlayer`.

Use this when you want:

- `MidiManager` → MPTK synth playback
- A "software output deviceId" that appears alongside real devices

MPTK → MidiManager adapter (source)

`MptkMidiManagerInputAdapter` can subscribe to MPTK callback events (commonly exposed by MPTK players) and inject matching MIDI 1.0-style events into `MidiManager`.

Use this when you want:

- MPTK playback / realtime generation to drive your existing `MidiManager` handlers
- MPTK to behave like a virtual *input* device

Convenience utilities

`MptkUtility` wraps common setup:

- Ensuring MPTK globals exist
- Creating MPTK players
- Creating + registering virtual devices
- Creating adapters
- Helper methods to inject MIDI events as virtual input

Sample code (ready-to-run)

These examples live under `Assets/MIDI/Samples/DocumentationExamples/`:

- **MIDI 1.0 → MPTK (virtual output sink)**
  `Assets/MIDI/Samples/DocumentationExamples/MptkVirtualOutputSinkExample.cs`
- **MPTK → MIDI 1.0 (inject into MidiManager as virtual input)**
  `Assets/MIDI/Samples/DocumentationExamples/MptkToMidiManagerInputExample.cs`

Sample scenes (integrated)

The built-in sample scenes also include optional MPTK integration:

- MIDI 1.0 sample scene script:
  `Assets/MIDI/Samples/Scripts/MidiSampleScene.cs`
  (adds a toggle to use an MPTK-backed virtual output device)

- MIDI 2.0 sample scene script:
  `Assets/MIDI/Samples/Scripts/Midi2SampleScene.cs`
  (adds a toggle to *mirror* MIDI 2.0 sends into an MPTK MIDI 1.0 virtual sink)

Choosing IDs, groups, and routing

Recommended convention:

- Use a clearly virtual prefix (e.g., `mptk:internal`, `virtual:sequencer`, `test:device`).
- Use `group = 0` unless you intentionally model multiple groups.

Because the virtual device appears in the device sets, you can build UI that lets users select it just like a hardware device.

Troubleshooting

“It compiles in Editor but fails on CI / another machine”

- Ensure `FEATURE_USE_MPTK` is only enabled when the MPTK asset is actually present.

“No events are received”

- Confirm the virtual device was registered as **input** (for injected events to look like input devices).
- Confirm your handler is registered with `MidiManager`.
- Confirm `deviceId` and `group` match what you expect.

“No sound from MPTK sink”

- Confirm the virtual device was registered as **output**.
- Confirm the sink’s `deviceId` matches the device you are sending to.
- Confirm MPTK global setup/resources are valid in your project (SoundFont / configuration).

Related docs

- MIDI 1.0 (MidiManager)
- Samples