

Platforms & Limitations

Feature matrix (by platform)

| Platform | Bluetooth MIDI | USB MIDI | Network MIDI (RTP-MIDI) | Nearby Connections MIDI | Inter-App MIDI | USB MIDI 2.0 | Network MIDI 2.0 (UDP MIDI 2.0) |
|---|----------------|----------|-------------------------|-------------------------|----------------|--------------|---------------------------------|
| iOS | ○ | ○ | ○ | ○ | ○ | ○ | △ (experimental) |
| Android | ○ | ○ | △ (experimental) | ○ | ○ | ○ | △ (experimental) |
| Universal Windows Platform | - | ○ | △ (experimental) | - | ○ | - | △ (experimental) |
| Standalone macOS / Unity Editor macOS | ○ | ○ | ○ | ○ | ○ | ○ | △ (experimental) |
| Standalone Linux / Unity Editor Linux | ○ | ○ | △ (experimental) | - | ○ | ○ | △ (experimental) |
| Standalone Windows / Unity Editor Windows | - | ○ | △ (experimental) | - | ○ | - | △ (experimental) |
| WebGL | ○ | ○ | - | - | - | - | - |

Legend:

- ○ supported
- △ supported but experimental / limited
- ◦ not supported

Limitations / requirements

Android

- USB MIDI: API Level 12 (Android 3.1) or above.
 - Bluetooth MIDI: API Level 18 (Android 4.3) or above.
 - Inter-App MIDI: API Level 23 (Android 6.0) or above.
 - MIDI 2.0: API Level 23 (Android 6.0) or above.
 - If you build with **Mono**:
 - latency issues may occur
 - may only support `armeabi-v7a`
 - Recommended: use **IL2CPP**
- `Project Settings > Player > Configuration > Scripting Backend`
- Nearby Connections MIDI:
 - requires API Level 28 (Android 9) or above
 - and should be compiled with API Level 33 (Android 13) or above

Bluetooth MIDI Peripheral mode (Android only)

In addition to acting as a BLE MIDI **Central** (scanning/connecting to devices), Android can also act as a BLE MIDI **Peripheral** (advertise your app as a MIDI device).

Notes:

- This is Android-only.
- See [MIDI 1.0 \(MidiManager\)](#) for the API entry points.
- Platform permissions and Bluetooth state still apply (see [Build PostProcessing](#)).

Meta Quest (Oculus Quest)

Meta Quest devices run Android, so the **Android** requirements above apply.

Practical notes (mirrors the manual):

- **USB MIDI** on Quest may require enabling a USB device intent filter during Android build postprocessing.
 - See: [Build PostProcessing & Scripting Define Symbols](#) (Quest USB MIDI section)
- **Bluetooth MIDI** device discovery/pairing can use Android's Companion Device workflow.
 - Enable scripting define symbol: `FEATURE_ANDROID_COMPANION_DEVICE`
 - See: [Build PostProcessing & Scripting Define Symbols](#)

iOS / macOS

- Supported iOS: 12.0 or above.
- Bluetooth MIDI: Central mode only.

UWP

- Supported UWP target version: 10.0.10240.0 or above.

- MIDI 2.0 (USB) is currently not supported.
- Bluetooth MIDI is not supported.
- RTP-MIDI requires enabling capability:
 - `Project Settings > Player > Capabilities > PrivateNetworkClientServer`

Windows

- Bluetooth MIDI is not supported.
- MIDI 2.0 (USB) is not supported.

Linux

- When MIDI 1 and MIDI 2 coexist on one USB device, only the MIDI 2 port may be found.

WebGL

- Device support depends on OS/browser WebMIDI environment.
- WebGL can't use raw UDP/TCP sockets, so RTP-MIDI / UDP MIDI 2.0 are unavailable.
- WebGL may have restricted `UnityWebRequest` access to other servers; use `StreamingAssets` for `.mid` and other content.
- WebGL cannot handle USB MIDI 2.0 devices nor network MIDI 2.0; MIDI 2.0 runtime functions are not available except clip file read/write.
- WebGL template may need to expose `unityInstance` (see [Transports & Platform Notes](#)).