MPE (MIDI 多维多音列表达)

本页面介绍了由以下文件提供的 MPE 支持：

- `Assets/MIDI/Scripts/MpeManager.cs`
- `Assets/MIDI/Scripts/IMpeEventHandler.cs`

MPE 是作为 MIDI 1.0 风格消息和 `MidiManager` 之上的一个层实现的。

本层的功能

输入 (MPE → 统一回调)

- 监听通过管理通道（下部区域为通道 0，上部区域为通道 15）上的控制变更 (CC) 发送的 **MPE 配置消息 (MCM)**。
- 构建包含**下部/上部区域 (Lower/Upper Zones)**、成员通道及区域状态的内部模型。
- 当成员通道收到事件时，将其作为 **MPE 事件**转发给 MPE 特定的处理器接口，并将**区域管理通道**作为区域标识符进行报告。

输出 (统一调用 → MPE 通道分配)

- 提供 `SendMpeNoteOn` / `SendMpeNoteOff` 等方法，这些方法会：
  - 为每个音符选择合适的成员通道，
  - （可选）重用最近使用的通道，
  - 跟踪每个通道的活动音符，
  - 在播放新音符之前应用区域范围内的参数。

区域与通道（概念）

- **下部区域 (Lower zone)**：管理通道 = 0，成员通道 = 1..N
- **上部区域 (Upper zone)**：管理通道 = 15，成员通道 = (15-N)..14

一个设备可以配置其中一个或两个区域，但两个区域的总成员通道数必须符合 MIDI 通道空间（共 16 个通道）。

输出 API：MpeManager

`MpeManager` 是一个单例风格的管理器（`MpeManager.Instance`），提供以下功能：

区域设置

- `SetupMpeZone(deviceId, managerChannel, memberChannelCount)`

该方法会：

- 验证 `managerChannel` 是否为 `0` 或 `15`。
- 将 `memberChannelCount` 限制在 0..15 范围内。
- 更新内部区域状态。
- 通过 `MidiManager` 向设备发送 MPE 配置 RPN/CC 序列。

MIDI 模式变更

- `ChangeMidiMode(deviceId, channel, mode)`

支持的模式：

- `3`: Omni Off, Poly
- `4`: Omni Off, Mono

（其他值将被忽略。）

发送事件

- `SendMpeNoteOn(deviceId, channel, note, velocity)`
- `SendMpeNoteOff(deviceId, channel, note, velocity)`
- `SendMpePolyphonicAftertouch(deviceId, channel, note, pressure)`
- `SendMpeControlChange(deviceId, channel, function, value)`
- `SendMpeProgramChange(deviceId, channel, program)`
- `SendMpeChannelAftertouch(deviceId, channel, pressure)`
- `SendMpePitchWheel(deviceId, channel, amount)`
- 此外还提供用于 SysEx 和系统消息的便捷封装。

通道选择行为（摘要）

当目标设备启用 MPE 时：

- 音符事件会被路由到选定的**成员通道**。
- 如果同一个音符已在某个成员通道上播放，管理器可能会在重新播放前先发送 Note Off 以停止该音符。
- 在开始播放音符前，会将区域范围内的参数应用到选定的通道：
  - 控制器
  - 多音触后 (Poly Aftertouch) 缓存
  - 程序 (Program)
  - 通道触后 (Channel Aftertouch)
  - 弯音 (Pitchbend)

如果未配置 MPE，`MpeManager` 将退回到直接发送到指定通道的模式。

输入 API：MPE 事件处理器

当已知区域配置时，传入的 MIDI 1.0 消息会被解释为 MPE。

典型的 MPE 处理器接口包括：

- `IMpeNoteOnEventHandler`
- `IMpeNoteOffEventHandler`
- `IMpeControlChangeEventHandler`
- `IMpePitchWheelEventHandler`
- `IMpeZoneDefinedEventHandler`
- 等等。

输入层会报告：

- `deviceId`
- **区域管理通道**
- 事件参数（音符、力度、控制器/值等）

实践用法模式

1. 为您的输出设备配置 MPE：

- 调用 `MpeManager.Instance.SetupMpeZone(...)`

2. 使用 `SendMpeNoteOn/Off` 而不是 `MidiManager.SendMidiNoteOn/Off` 来发送音符。
3. 实现 `IMpeZoneDefinedEventHandler` 以响应传入的区域变更（在设备配置动态变化时非常有用）。

源代码示例实现 (MPE 输出)

请参考：

- `Assets/MIDI/Samples/Scripts/DocumentationExamples/MpeOutputExample.cs`

将其附加到 GameObject 上并进入播放模式。
它将执行以下操作：

- 初始化 MIDI 1.0
- 配置下部区域 MPE 布局
- 使用 MPE 通道分配发送测试音符