How to use geekdrums.MusicEngine

https://github.com/geekdrums/MusicEngine

Music Programmer

@geekdrums

MusicEngineとは

Unityで音楽同期演出やインタラクティブミュージックを実現できます。

https://github.com/geekdrums/MusicEngine

から無料でDLできます。
MITライセンスで公開してあります。

商用含め利用は自由ですが、不具合などがあってもサポートや修正をお約束はできませんのでご承知おきください。

利用に際してご連絡も不要ですが、お役に立てたようでしたらTwitter:@geekdrumsまでご一報ただけると嬉しいです。

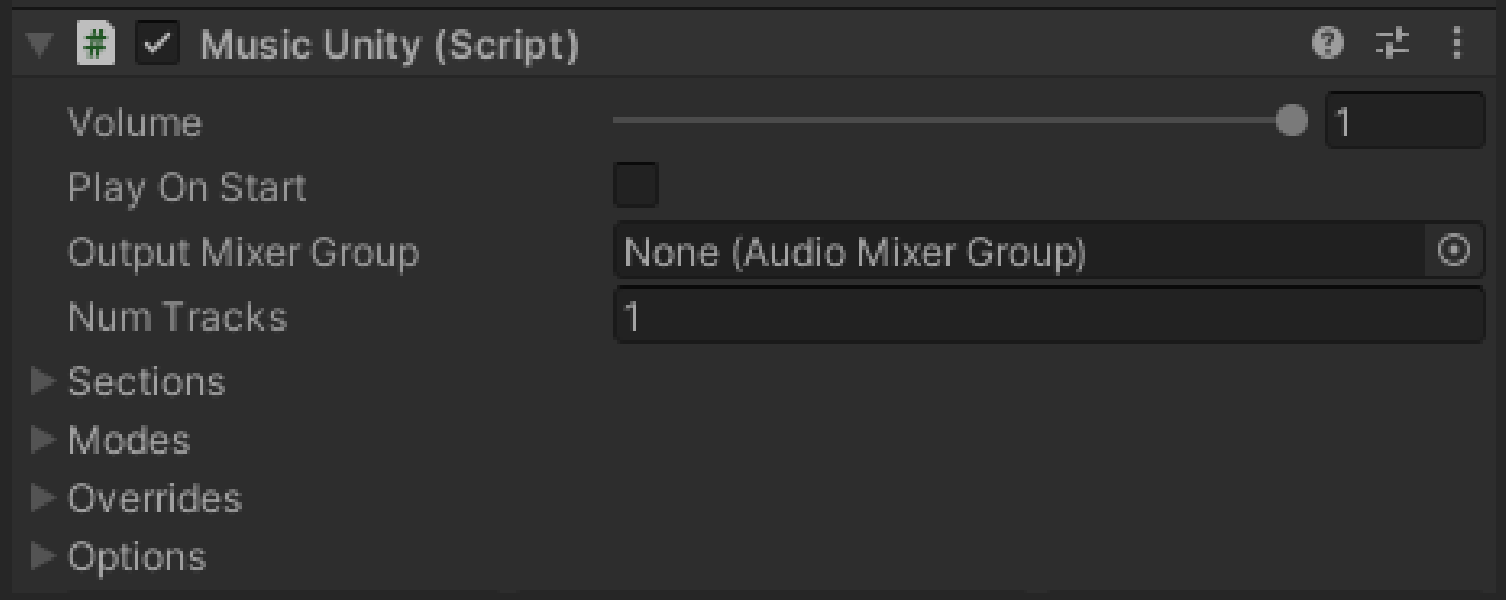
1,利用準備

1, Music Unityを追加

Play On Start：最初から再生する場合はチェック

Output Mixer Group：指定は任意です

Num Tracks：とりあえず1でOKです



2, Sectionを定義

最低一つ以上のSectionを定義します。

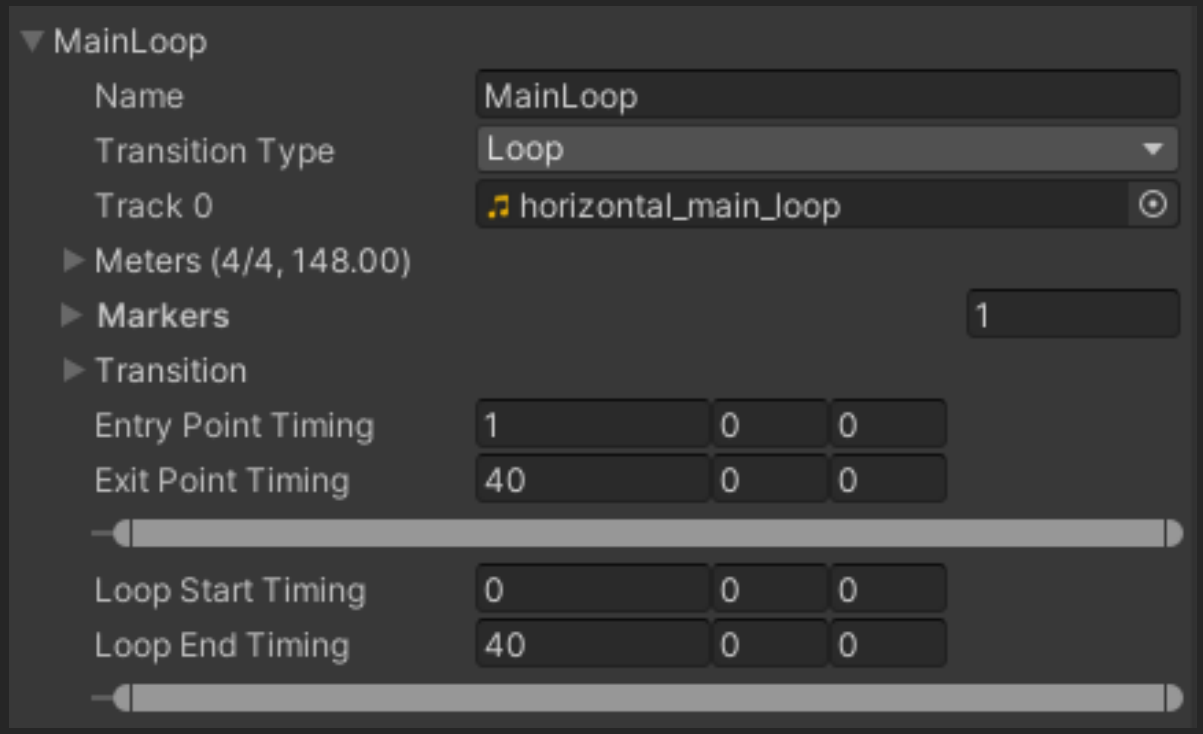
Track 0にセクションの波形（Audio Clip）を指定してください。

Transition Type：

- Loop：そのセクションをループ
- Transition：自動的に別セクションに遷移
- End：そのセクションで終わり

から選べます。

ループする場合、Loop Start/Endのタイミングをそれぞれ指定してください（Loop StartはEntry Pointからの相対値で、Entry Pointより後にしか設定できません）。



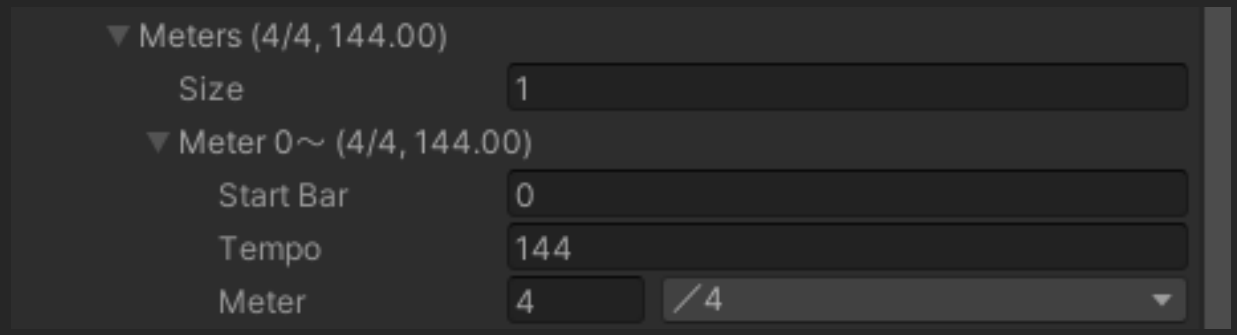
3, Meterを定義

Metersに一つ以上の拍子情報を設定してください。

Start Bar：Start Bar=0のものが一つあればOK。途中から拍子やテンポが切り替わる場合、これを指定してMetersを増やしてください。

Tempo：テンポを指定してください。

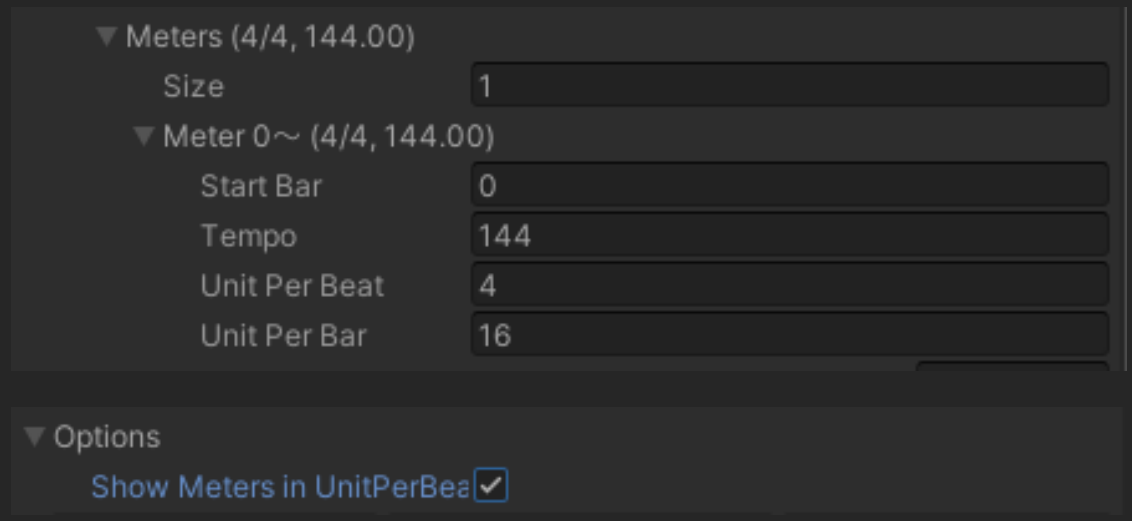
Meter：拍子を登録してください。分母の方は/4,/8,/16から選べます。



(Optional) 複雑な拍子を扱う場合

デフォルトでは、1拍＝16分音符4個分としています。例えばMeterで6/8を指定しても、1拍=2/8と解釈されます。

これを変更したい場合、Optionsの**Show Meters in UnitPerBeat**にチェックを入れると、Meterの表示が4/4など簡略的なものから詳細なものに切り替わります。



Unit Per Beat： 何unitを1拍と捉えるか

Unit Per Bar：何unitを1小節と捉えるか

これを指定することで、任意の細かさ（3連付、6連付、5連付など）で拍を割ったり、任意の長さの小節を設定できます。

5, Music.Play(name)

Musicはstaticクラスで、**Music Unity**(or Music ADX2 / Music Wwise)**インスタンスのうち現在再生中のもの**一つだけを常にCurrentとして保持しています。

Music.Play(name)関数を利用することで、再生する音楽を指定できます。name引数には、Hierarchy上で指定したGameObject名を指定します。

あるいは、Play On Startを指定していれば自動的にMusic.Currentに反映されます。

音楽同期演出やインタラクティブミュージックは、基本的にこの**MusicというstaticクラスのAPIで実現**できます。

```
void ResetGame()
{
    if( gameState_ != GameState.InGame )
    {
        if( Music.IsPlaying )
        {
            Music.Stop();
        }
        Music.Play("Music");
    }
```

2,音楽同期演出

Music.IsJustChanged〇〇

Update関数内でIsJustChanged〇〇系のAPIを使えば、音楽的なタイミングに合わせて演出をトリガーできます。

- IsJustChangedBar() : 小節ごと
- IsJustChangedBeat() : 拍ごと
- IsJustChangedAt(Timing) : 指定タイミング

```
if( (gameState_ == GameState.Result || IsReplaying) && Music.IsJustChangedBar() )
{
    NewGameButtonAnim.Play();
}
```

Music.Just / Near

音楽時間を表すTimingクラスは

(Bar, Beat, Unit)

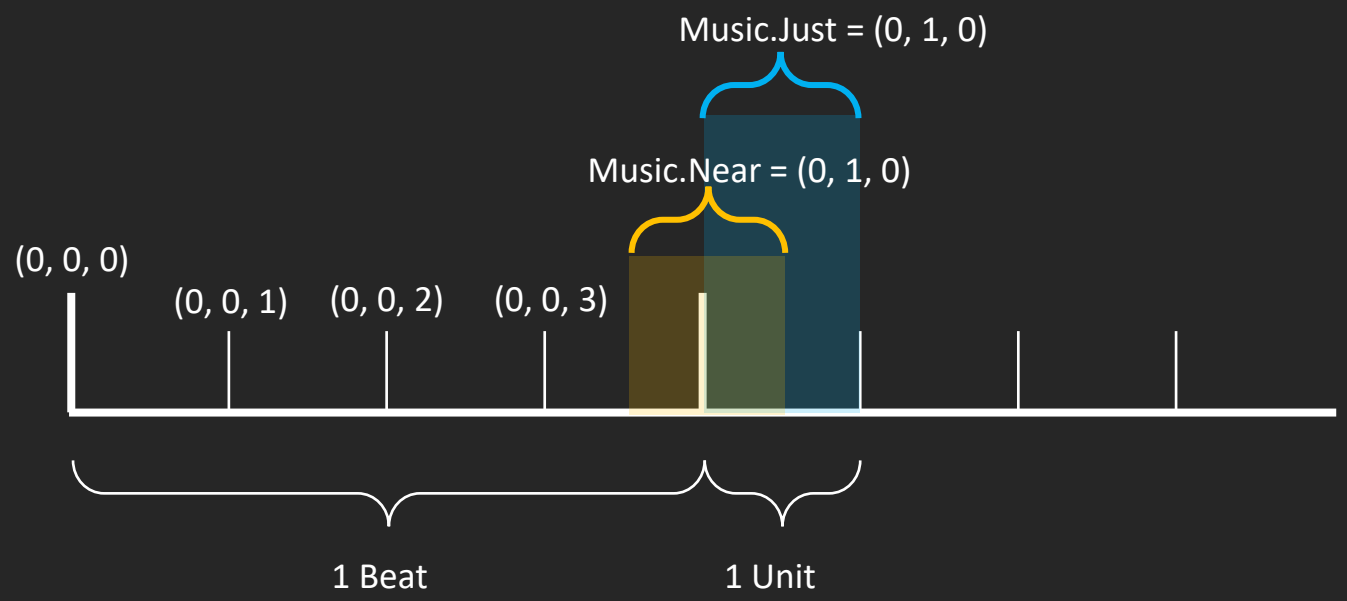
の情報から構成されます。それぞれ、

（小節, 拍, 16分音符※）

の解像度で音楽時間を表しています。

※UnitPerBeatの設定がデフォルトの4であればそのとおりだが、
ここを変更すると違う単位にもできる

Timing型の
Music.Justは「経過した最後のタイミング」を、
Music.Nearは「最も近いタイミング」を
指し示しています。



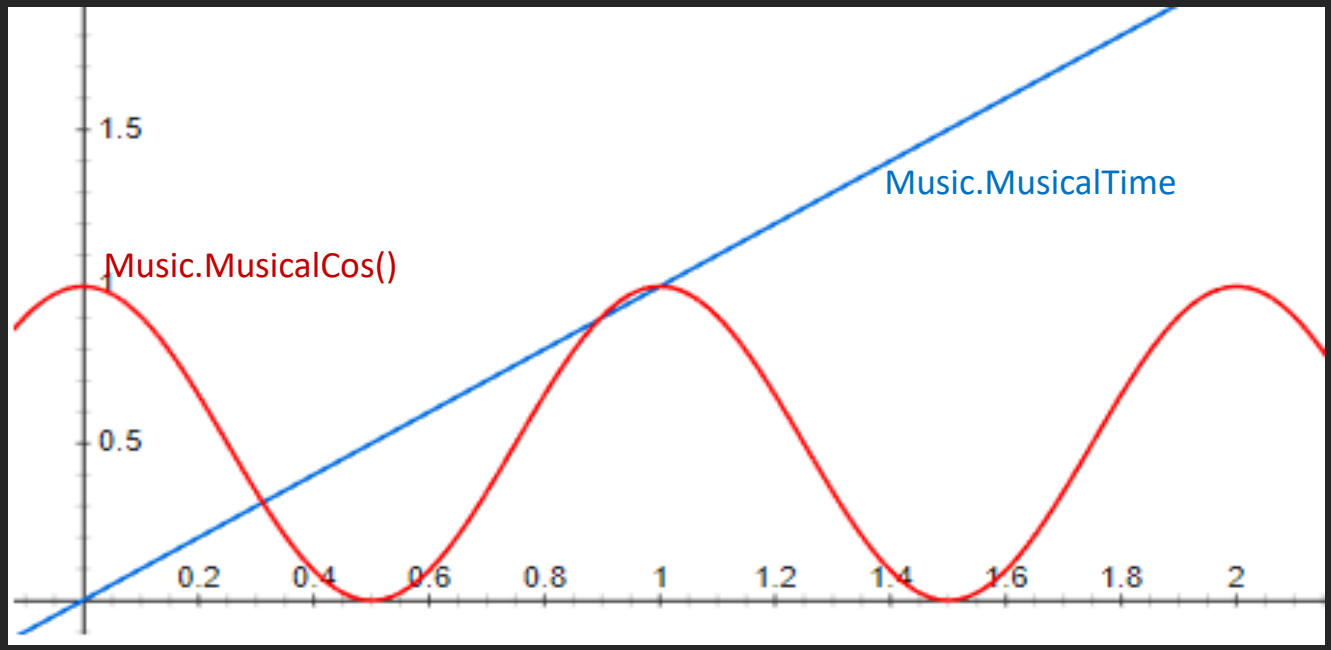
Music.MusicalTime

Music.MusicalTimeは、小節数※に合わせて連続的に変化するfloat値を取得できます。

Timing型のMusic.Justなどでは整数値しか取れないので、音楽に合わせた連続的な変化の演出にはこちらを活用してください。

※もし、途中で別のMeterが定義されてテンポや拍子が変更された場合は、それも加味したうえで小節数を正規化して返します。つまり、MusicalTimeの変化速度も音楽に合わせて動的に変化します。

このMusicalTimeをコサイン波に変換したものを個人的によく使うため、MusicalCosというAPIを用意しています。デフォルト引数では右図の赤線のような変化をします。



音楽時間(小節)→ 1小節 2小節

```
// 4小節で0から1になる変数
float normalizedTime = (Music.MusicalTime / 4.0f);
```

3,インタラクティブミュージック

縦の遷移 / アレンジの変化

楽器やアレンジごとにレイヤーを分けて、音量変化で遷移

参考：
ゲーム音楽をイロイロに彩る「アレンジの変化」｜じーくどらむす｜note
https://note.com/geekdrums/n/need52e4a7344

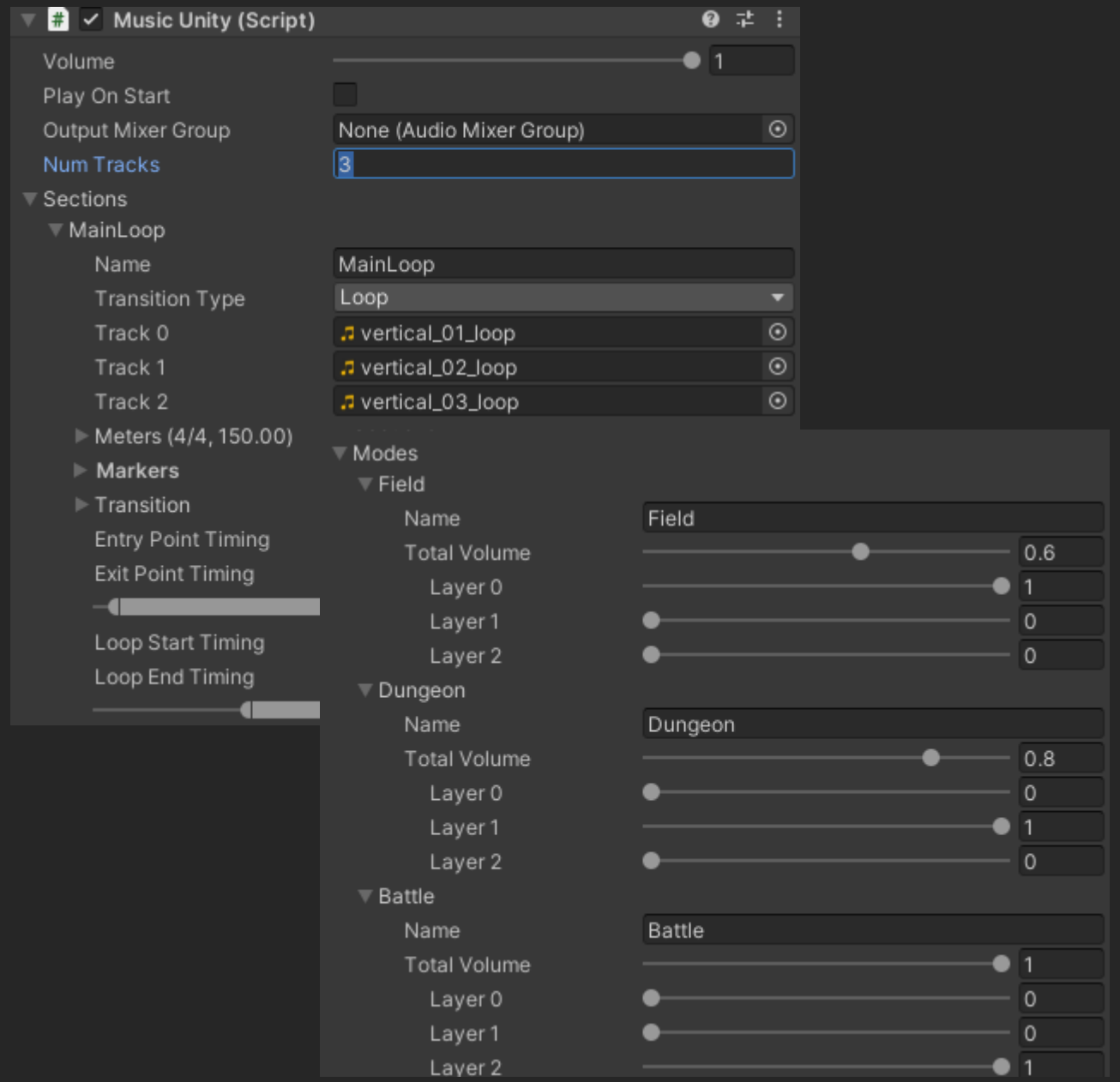
縦の遷移の準備

Num Tracksに同時再生する最大数を入力します。

各セクションのTrack 0 ～ にそれぞれ楽器ごとやアレンジごとの波形を追加してください。

このままではすべてのレイヤーが同時再生されてしまうため、指定したレイヤーだけを再生するために、**Mode**というものを定義します。

各モードに対して、各レイヤーボリュームと全体ボリュームをそれぞれ設定できます。このモードを切り替えることで縦の遷移を実現します。



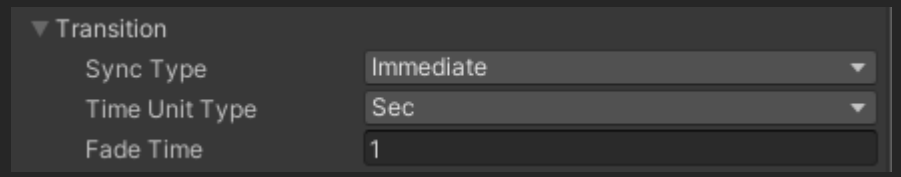
縦の遷移の実行と調整

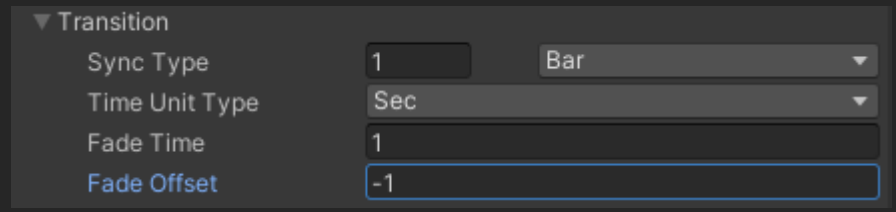
Music.SetVerticalMix(name)に、モード設定でつけたモードの名前を指定することで遷移が実行されます。

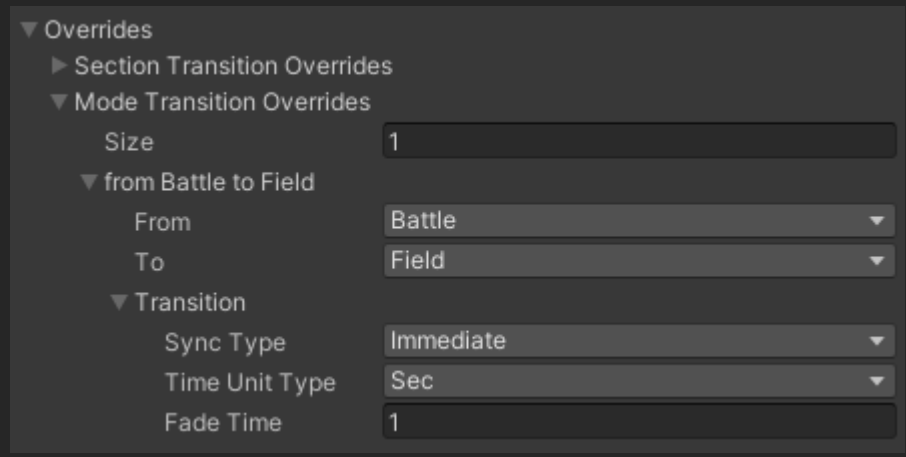
このとき、クロスフェードのタイミングや秒数はModesの下に表示されるTransitionツリー内で指定できます。

SyncTypeをImmediate（即時）以外にすると、小節などに合わせてフェードを開始できます。
FadeOffsetをマイナスにすることで、アウフタクトからフェード開始して小節頭でフェード完了させることもできます。

さらに、特定のモード遷移でトランジションを別にしたい場合、Overrides以下のMode Transition Overridesに遷移元と遷移先を指定したうえでTransitionを定義することで上書きが可能です。







横の遷移 / 展開の変化

展開ごとにセクションを分けて、それぞれをループ

次の小節など、音楽的に繋がる遷移タイミングを待ってから遷移

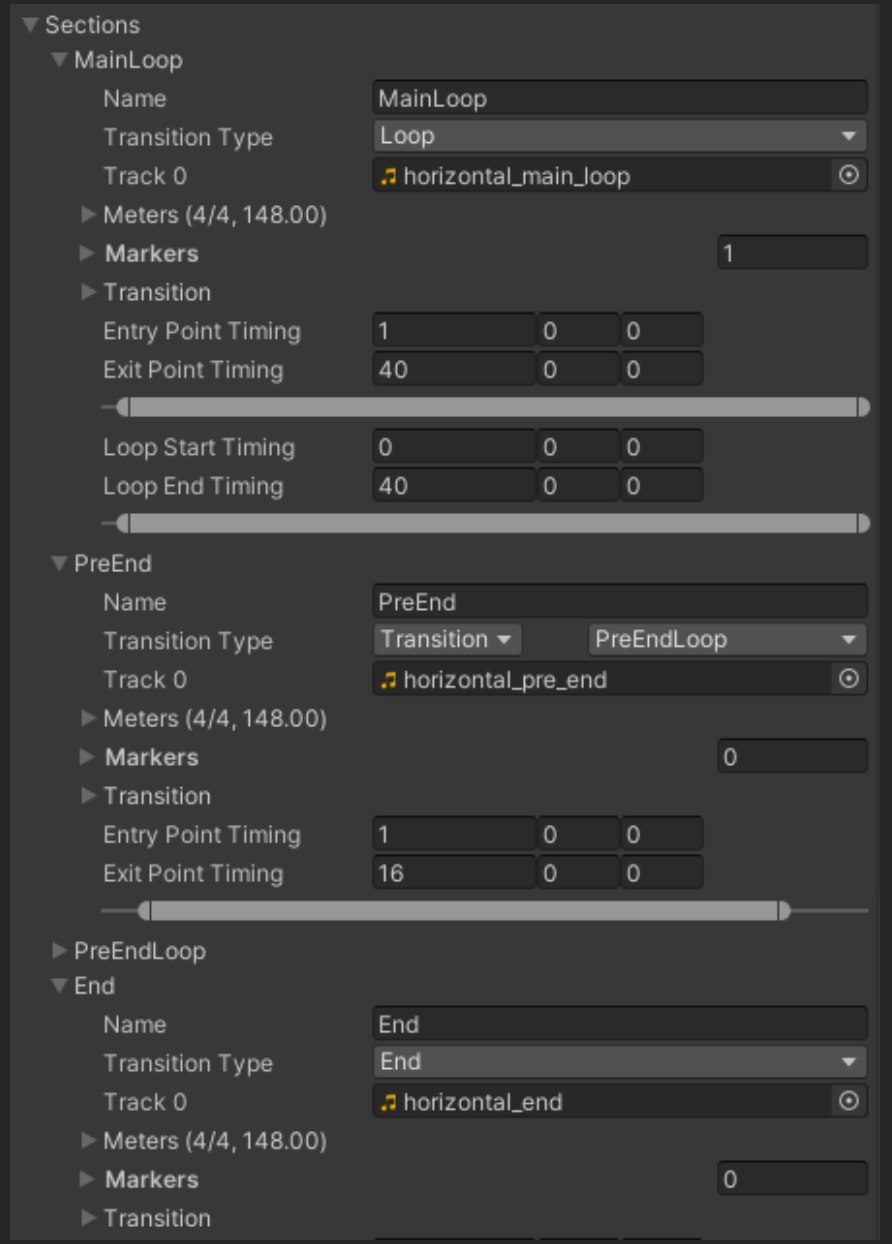
参考：
ゲーム音楽が「ループ」をやめる時｜じーくどらむす｜note
https://note.com/geekdrums/n/n035a57e23a8a

横の遷移を使うには

SectionsのAddボタンを押してセクションを増やします。

セクションごとにループする波形（あるいは終わり用や遷移用の波形）を登録し、それぞれに「1,利用準備」のところで説明したようにTransition TypeやLoop Start/Endを設定します。

右図では、MainLoopはループして、そこからPreEndに遷移した場合はPreEnd再生後に自動的にPreEndLoopに遷移し、最後にEndに遷移した場合は楽曲が終了するという作りになっています。



横の遷移の実行と調整

Music.SetHorizontalSequence(name)に、セクション設定でつけた名前を指定することで遷移が実行されます。

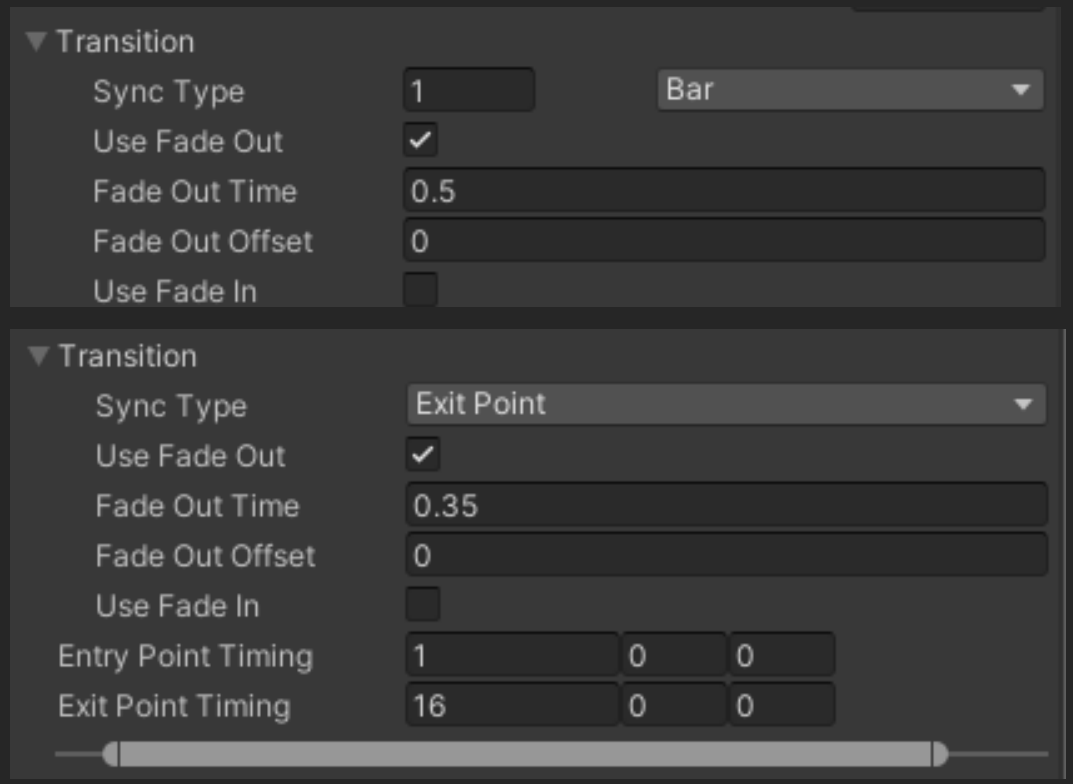
セクションごとにある「Transition」ツリーでは、このセクション「から」他のセクションへ遷移する際の設定を編集できます。

Sync Type：デフォルトでは波形をExit Pointになっていますが、小節ごと（Bar）、拍ごと（Beat）、その数（2小節ごととか）も指定できます。

Use Fade In/Out：遷移時にフェードインアウトを指定したい場合に利用してください。

Entry / Exit Point：遷移時にアウフタクトで重ねてほしい部分はEntry Pointより前に、遷移時にリリース部分として重ねてほしい部分はExit Pointより後になるように設定してください。

また、Overrides以下のSection Transition Overridesを追加することで、特定のセクション間の遷移設定を上書きできます。



geekdrums@gmail.com

Twitter: @geekdrums

おわり

ご質問やご感想は上記まで。
前述の通り、不具合などがあってもサポートや修正を
お約束はできませんのでご承知おきください。