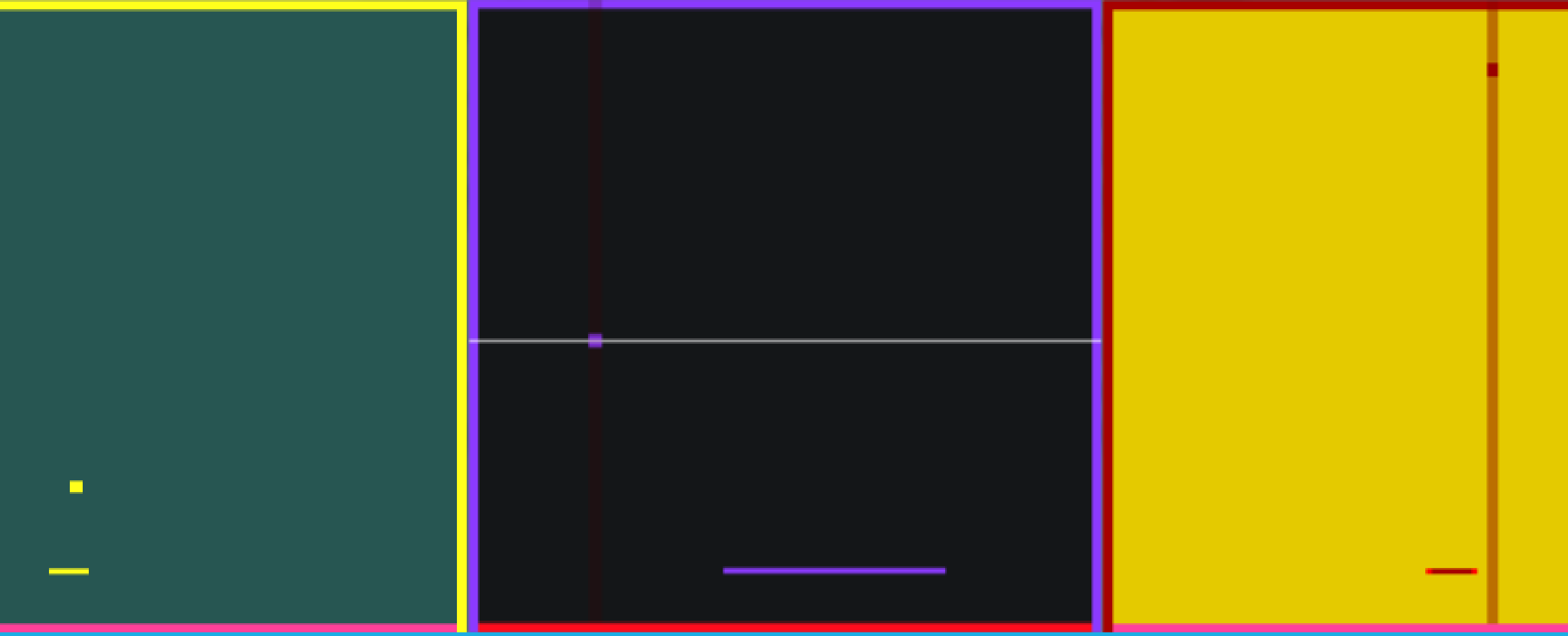
MusicEngine for Unity/ADX2LE

GEEKDRUMS

Examples

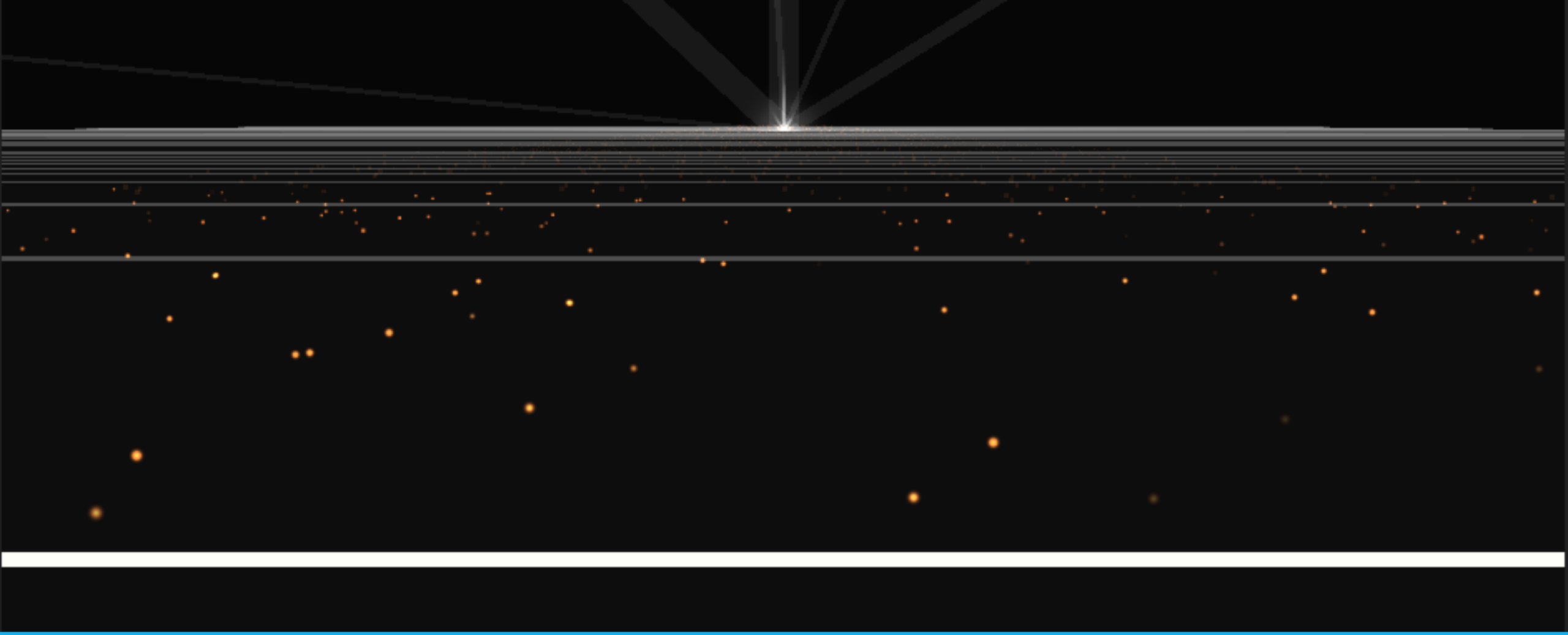
Games I made with MusicEngine



MusicPong

http://unitygameuploader.jpn.org/game/1233.html

Included in MusicEngine as an Example Project.



Space to go

http://www.ludumdare.com/compo/ludum-dare-29/?action=preview&uid=25923

LudumDare #29 gold medal on Audio category(compo)

VOXQUARTER

http://voxquest.tumblr.com/

Now in development

Need a library that can detect “Musical Time”

Intelligent Music System for everyone.

What is MusicEngine

A single MonoBehaviour script. → Music.cs

Things you can do

- easily access to What bar/beat/unit is it now?
- easily animate With Synced to Music.

Things you can NOT do

- animate with synced to audio data: use **GetSpectrumData** instead.
- change music dynamically: use **ADX2LE/WWise** instead.

DOWNLOADS

https://github.com/geekdrums/MusicEngine

Update

geekdrums authored 11 hours ago — latest commit 6527dce184

Example	Update	11 hours ago
Music.cs	Update	11 hours ago
README.md	Update README.md	23 hours ago

You can see MusicPong(whole Unity project) inside Example folder.

You can use Music.cs alone.

or ADX2LE version is here↓

https://github.com/geekdrums/MusicEngineForADX

Premise

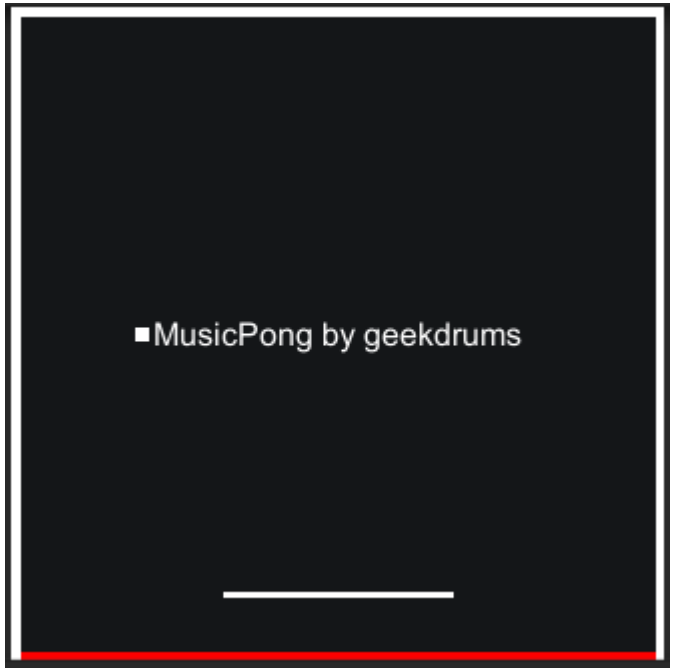
- Make music for your own.
  - MusicEngine can’t auto detect your music tempo/meters.
- You always have ONE Music.
  - You can always access to musical info from Music.something(static members)
- 1MusicalTime=sixteenth note(note)
  - Music.**MusicalTime** increases 1 per 1 sixteenth note.
  - (note)You can modify this unit depends on musics or sections.

How to make MusicPong

Music Pong | UnityGameUploader
http://unitygameuploader.jpn.org/game/1233.html

Make Pong.

SINGLE PLAYER PONG



Make Music

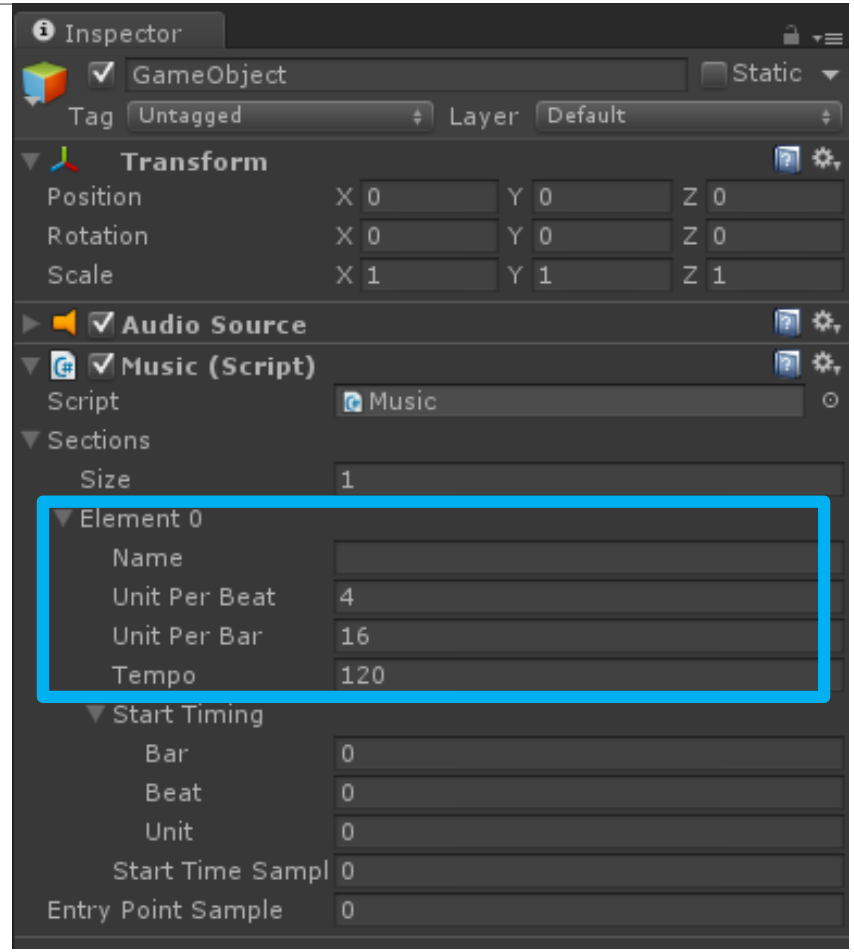
https://soundcloud.com/geekdrums/musicpong

5 steps to get things done.

- Step1. Add Music component
- Step2. Quantize sound
- Step3. Fit int the beat
- Step4. Animate along the beat
- Step5. Scene transition with the music

Step1. Add Music component

- Drag Music.cs to a gameobject
  - default section info will be added (tempo=120, 4/4)
- You are ready to access Music.Just from your code.



Note:

If you music doesn't starts from 0 sample,
please specify the EntryPointSample.

StartTimeSamples (inside the section info) will be automatically calcurated from Start Timing.

Tips1: Timing & Section

- class Timing
  - int Bar;
  - int Beat;
  - int Unit; //sixteenth note

  Note: Starts from (0,0,0). Last timing of 4 bar, 4/4 section will be (3,3,3).

- class Music.Section
  - int UnitPerBeat=4; //how many unit per beat
  - int UnitPerBar=16; //how many unit per bar
  - int Tempo=120; //beat per minutes
  - Timing StartTiming; //section's start timing

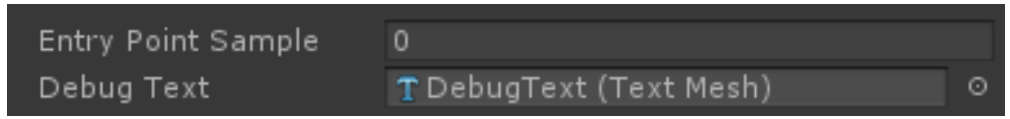
  Note: If you want to use 7/8, try UnitPerBeat=4 & UnitPerBar=14.

Ready

SYSTEM ALL GREEN

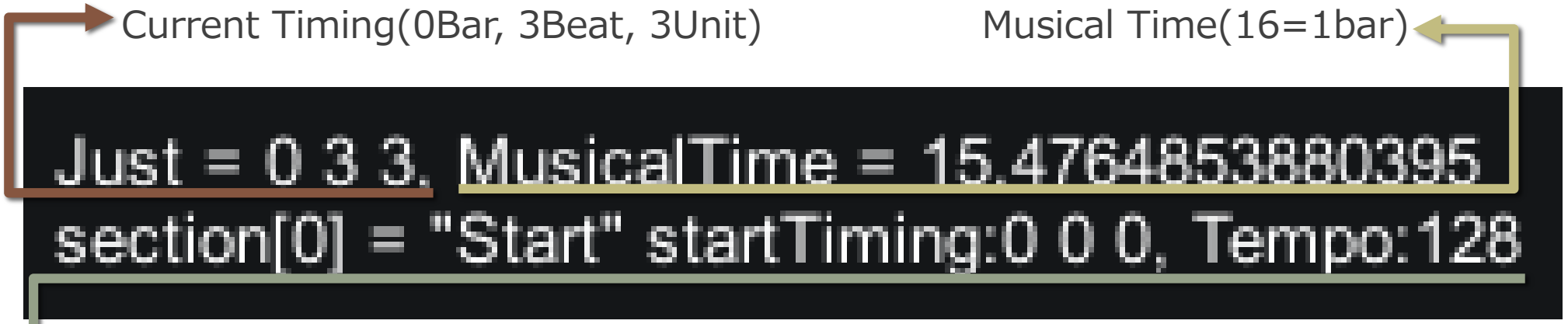
Tips2: DebugText

Attach 3DText(TextMesh) to DebugText→



Current Timing(0Bar, 3Beat, 3Unit)

Musical Time(16=1bar)



Current section. ”Start” section starts from (0,0,0), tempo = 128.

Note: You can suspend/resume or change pitch of music. It’s no probrem for timing mesuring.

Step2. Quantize sound

- Music.QuantizePlay(AudioSource source, int transpose);
  - Audio will be quantized to musical time.
  - “transpose” argument: 1 = half tone , 12 = octave.

    Ex. Ball.cs↓ Quantizing the reflect sounds.

```
        //side wall
        velocity.x = Mathf.Abs( velocity.x ) * -Mathf.Sign( transform.position.x );
        Music.QuantizePlay( GetComponent<AudioSource>() );
    }
    if( Field.FieldLength <= transform.position.y )
    {
        //roof
        velocity.y = Mathf.Abs( velocity.y ) * -Mathf.Sign( transform.position.y );
        Music.QuantizePlay( GetComponent<AudioSource>(), 7 );
```

Quantize + transpose. Simply fun.

GOOD COST PERFORMANCE!

Step3. Fit in the beat

- bool Music.IsJustChangedBar()/Beat()/At(Timing)
  - Turns true on every ONE FRAME the bar/beat/timing changes.

    Ex. Field.cs↓ Changing backgroundColor

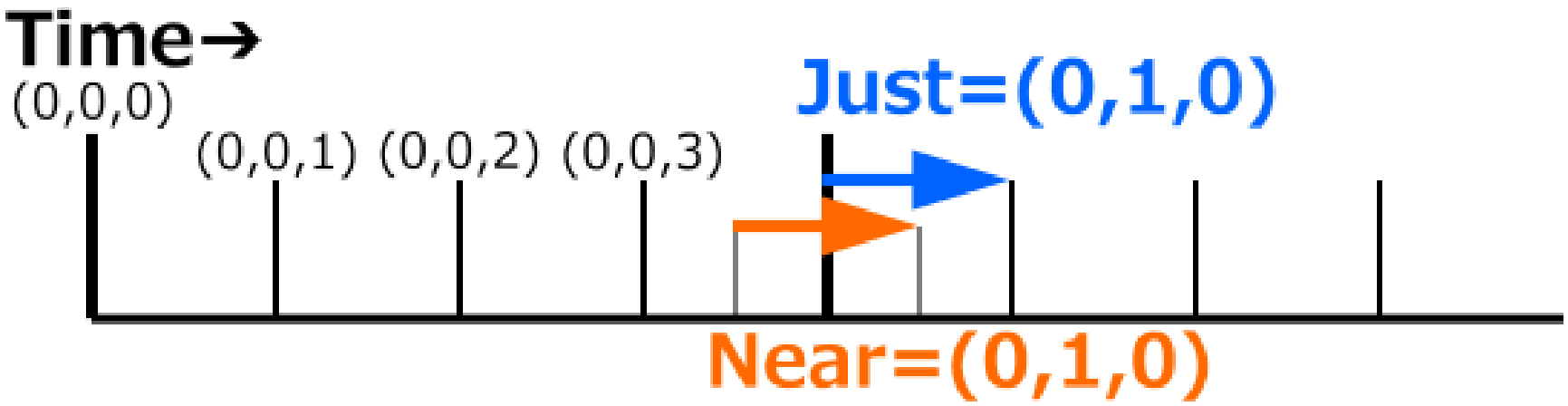
```
if( Music.IsJustChangedBar() )
{
    MainCamera.backgroundColor = Music.Just.Bar % 2 == 0 ?
        levels[currentLevel].BGColor : levels[currentLevel].BGChangeColor;
}
```

Music synced events.

BACK COLOR CHANGE LOOKS COOL.

Tips3: Just & Near

- Timing Music.Just : changes JUST after the beat.
- Timing Music.Near : indicates the NEARest beat.



Note: If the tempo=120 and you have 60 frame per sec, 1 MusicalTime has 7.5 frames in average.

Note: If you want to do something just BEFORE the timing, IsNearChanged event would be useful.

Step4. Animate along the beat

MUSIC SYNCED ANIMATION

Step4-1.Blink animation

- bool Music.IsFormerHalf
  - blink true/false in forer/later of a musical time.
- bool Music.IsJustChanged
  - Turns true every one frame of sixteenth notes.

Ex. Paddle.cs↓ damage blink animation.

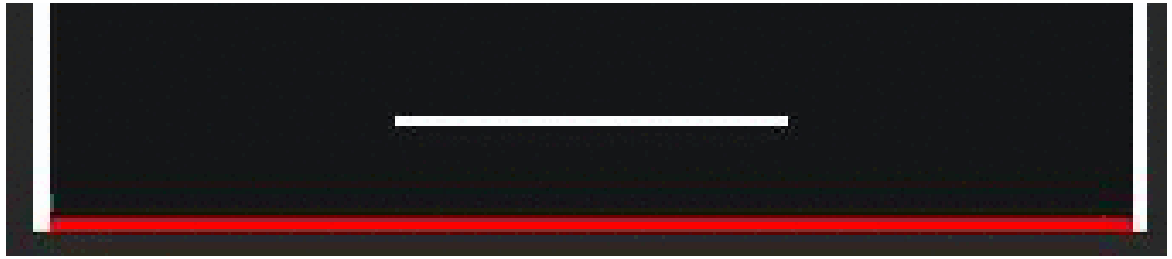
```
//damage
if( damageMusicalTime > 0 )
{
    redBar.GetComponent<Renderer>().material.color = Music.IsFormerHalf ? Color.red : Color.white;
    if( Music.IsJustChanged )
    {
        --damageMusicalTime;
```

Step4-2. Linear animation

- float Music.MusicalTime
- float Music.MusicalTimeFrom(Timing)
  - Get float time of music.

Ex. Padddle.cs↓ paddle showing animation in the beginning.

```
if( Music.CurrentSection.Name == "Start" )
{
    transform.localScale = new Vector3( initialScale.x * Mathf.Clamp01( (float)Music.MusicalTime / 16.0f ),
}
```



Step4-3. Swing animation

- float Music.MusicalCos(cycle, offset, min, max)
  - return cos value that swings from max=1 to min=0.

Ex. Field.cs↓ field floor & wall color animations.

```
 = Color.Lerp(EndBarColor, EndBarLerpColor, Music.MusicalCos(lerpMusicalTime));
:Level].materialColor, levels[currentLevel].lerpMaterialColor, Music.MusicalCos(lerpMusicalTime));
```

Step5. Scene transition with the music

EASY TO TRANSITION

Step5-1. Seek music to change scene.

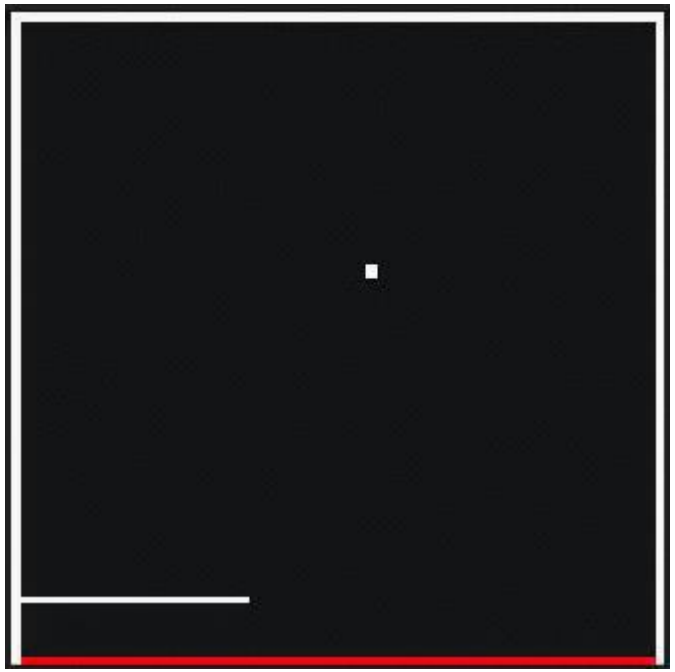
- Music.Seek(Timing)/SeekToSection(string name)
  - Seek to any musical timing.

Ex. Ball.cs↓ Game over logic.

```
else if( transform.position.y <= -Field.FieldLength )
{
    //floor
    Music.SeekToSection( "GameOver" );
}
```

Ex. Field.cs↓ Restart logic.

```
void Restart()
{
    Music.SeekToSection( "Start" );
    Music.Play( "Music" );
    ball.OnRestart();
    paddle.OnRestart();
```



Step5-2. Change scene by section info.

- Music.CurrentSection
  - Get current section info.

Ex. Field.cs↓I used section name as a scene.

```
switch( Music.CurrentSection.Name )
{
case "Start":
    UpdateStart();
    break;
case "Clear":
    UpdateClear();
    break;
case "GameOver":
    UpdateGameOver();
    break;
```

Done.

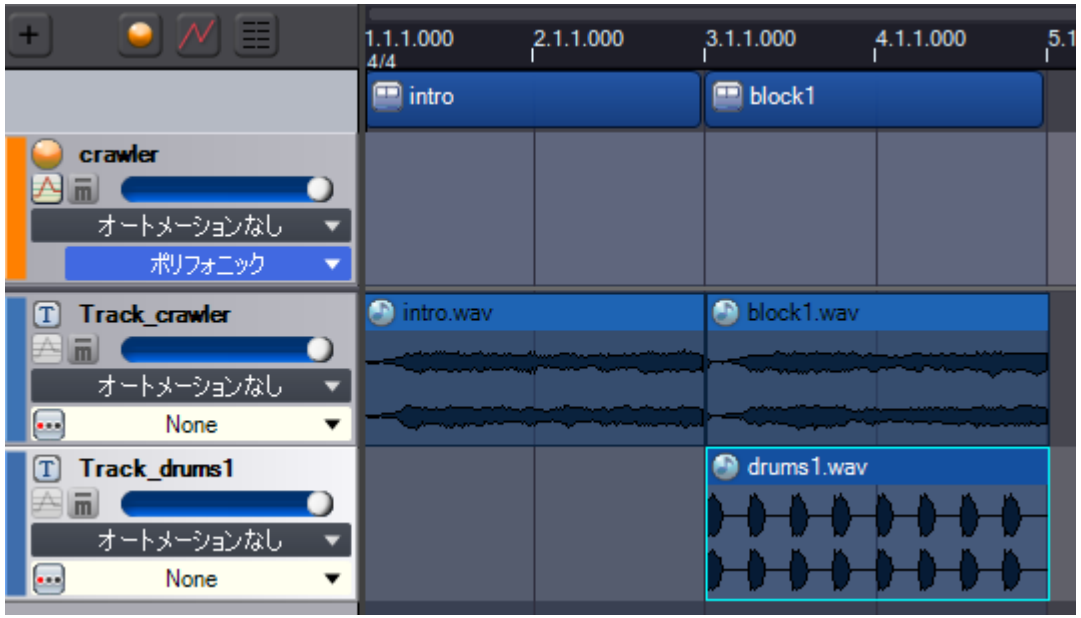
MUSIC IS (KIND OF) A MASTER CLOCK OF THIS GAME.

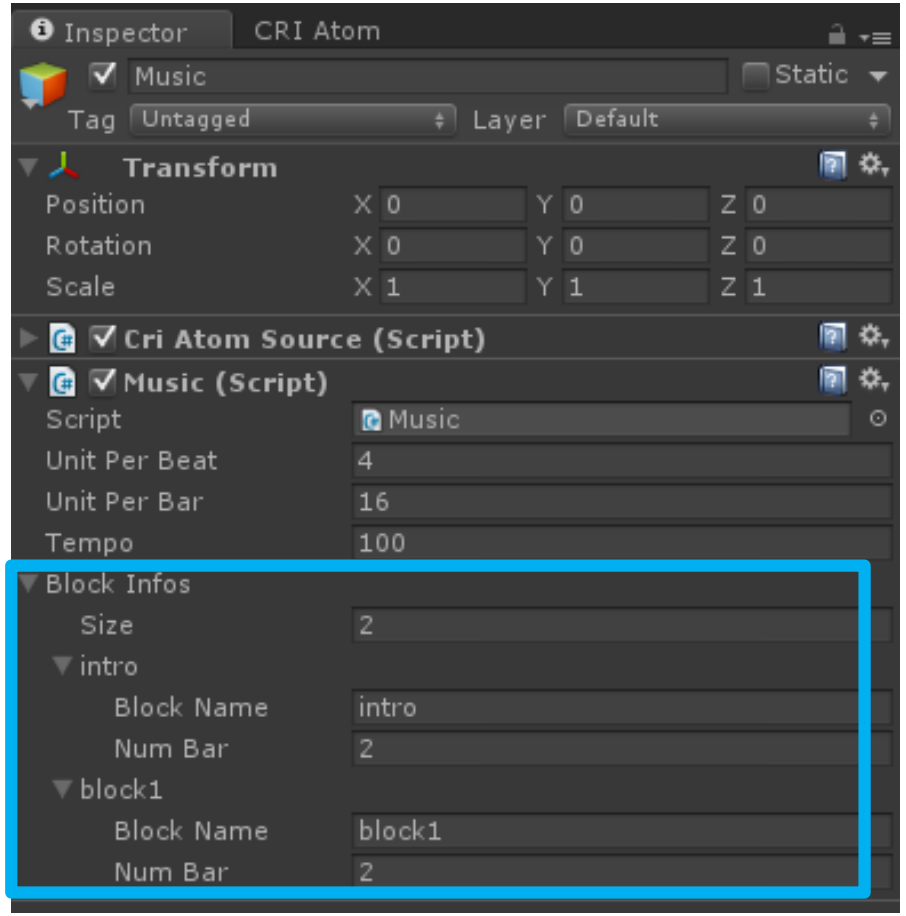
Difference between Unity+ADX2LE version & Unity StandAlone

IT’S ALMOST SAME.

Use Block instead of Section

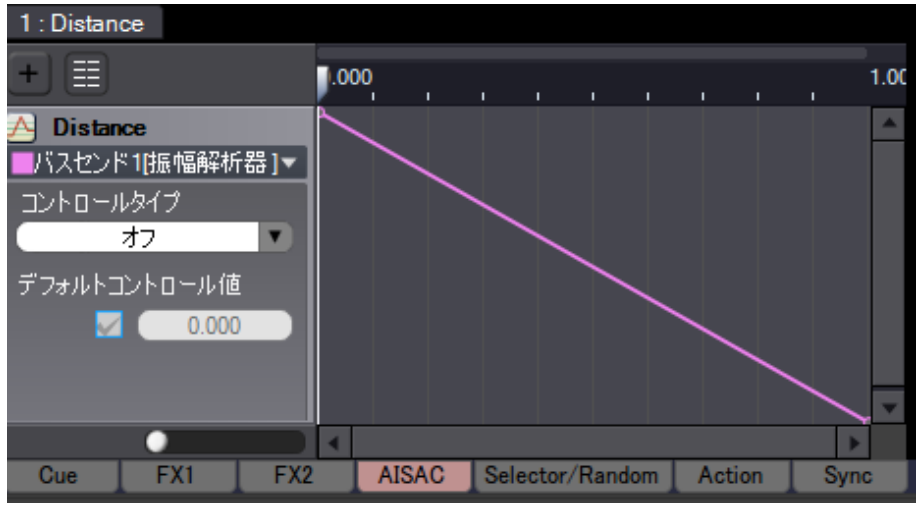
- Use “Block Resequencing” technique together.
  - No tempo & meter change inside a music.
  - Each block starts from (0,0,0) timing.
  - Fit the wave length to the block. like this.





Block resequencing, Aisac(RTPC)

- Music.Play(musicName, firstBlockName)
- Music.SetNextBlock(name/index)
- Music.SetAisac(name/index, value) →



That’s all.

```
//Copyright (c) 2014 geekdrums
//Feel free to use this for your lovely musical games :)
```