Sound Reactor®

Rhythm Game

Introduction

The trickiest part about making a rhythm game is getting the notes to emit in time with a song. Thankfully for us we can leverage an old file type called MIDI. Lots of tools exist out there that write MIDI files called MIDI sequencers or Digital Audio Workstations (DAW). With these tools we can overlay MIDI notes over audio in any pattern we like. Then the two can be played together in sync to create a rhythm game.

What you'll need

You'll need a few basic things before making a custom rhythm game with Sound Reactor Pro.

3rd Party Tools

- MIDI sequencer or DAW (not necessary for this guide)

Scripts

- RhythmInterface.cs
- RhythmEmitter.cs
- RhythmNote.cs
- AudioMidiSync.cs

Other Requirements

- Sound Reactor Pro
- Demo scenes packaged with Sound Reactor Pro
- Unity 5.5 or higher
- Basic knowledge of Sound Reactor

Important

Never use scripts found inside the Demos folder directly in production. Doing so will mean updates you make to those files will be lost if you import a newer release of the Unity package. To ensure the new file is unique, you must duplicate and then rename the file – renaming alone isn't enough.

Script Overview

Rhythm Interface

The **RhythmInterface** script keeps track of rhythm data, such as: hits, misses, and consecutive hits. Attach this script to any object you wish to use to update rhythm data. In the case of the RhythmGame.unity scene, **RhythmNote** points to and updates this data.

Rhythm Emitter

The **RhythmEmitter** reacts to MIDI note beats via Sound Reactor's **EventDriver**. Each beat (change in velocity) results in a **RhythmNote** being emitted. The emitter allocates a cache (size specified by the user) of notes and then cycles through them and repeats when the cache's size is reached. Caching keeps the garbage collector from causing hiccups at runtime.

Rhythm Note

The **RhythmNote** is emitted by the **RhythmEmitter**. This script handles timing and touch events. It is also responsible for updating rhythm data using **RhytmInterface**.

Connections

Following the layout of the RhythmGame demo scene, these are the connections between the scrips:

- **SoundReactorShort.midi.asset** connects to **MidiSource**
- **MidiSource** connects to **AudioMidiSync**
- **AudioMidiSync** connects to **RhythmInterface**
- **RhythmInterface** connects to **RhythmNote**

Important Values

Timing

Timing for the notes are controlled by the following values:

- **AudioMidiSync → Audio Delay:** The time it takes before the MIDI note appears to play in time with the audio. Cannot be adjusted at run time.
- **RhythmInterface → Timing Offset:** Fine adjustment to the timing for things like audio, video, and input lag.
- **RhythmNote → Hit Object and Miss Object:** Optionally emit an object at the position of the note for a hit or a miss.
- **RhythmNote → Debug:** Enable this to force objects to delete at the exact timing of the beats.

Speed

Speed for the notes are controlled by the following values:

- **Animation:** The distance the object is told to move in 1 second (the animation itself must be 1 second).
- **AudioMidiSync → Audio Delay:** The animation time is adjusted to match this time.

NOTE:

It takes time to get the timing just right. Use **RhythmNote → Debug** initially to place the BeatLine, and then make fine adjustments with the **Timing Offset** in the **RhythmInterface** to get touch timing down. After that, make tweaks to all the other values to get the notes where you want them to be during presentation.

MIDI file

The notes in the MIDI file are: C (note 72), C# (note 73), D (note 74) in the 5th Octave. Look for the FrequenceToMidi.pdf file under Documents to help you to convert MIDI frequencies to note values, or vice versa.

IMPORTANT:

If the timing of the MIDI is off during playback, it's most likely due to the MIDI file not having a tempo set. If this happens, set the Default Tempo on the ***.midi.asset*** file in the inspector and then hit the **Reload** button.

Contact

Direct all questions, suggestions, feature requests, and bugs to: support@littledreamergames.com