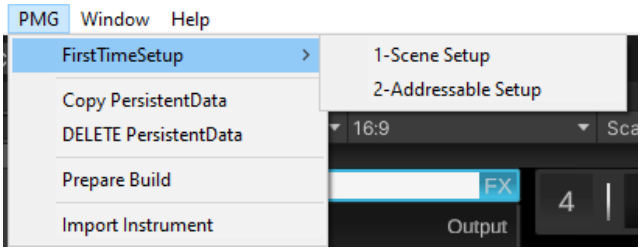
Video tutorials are available on Youtube

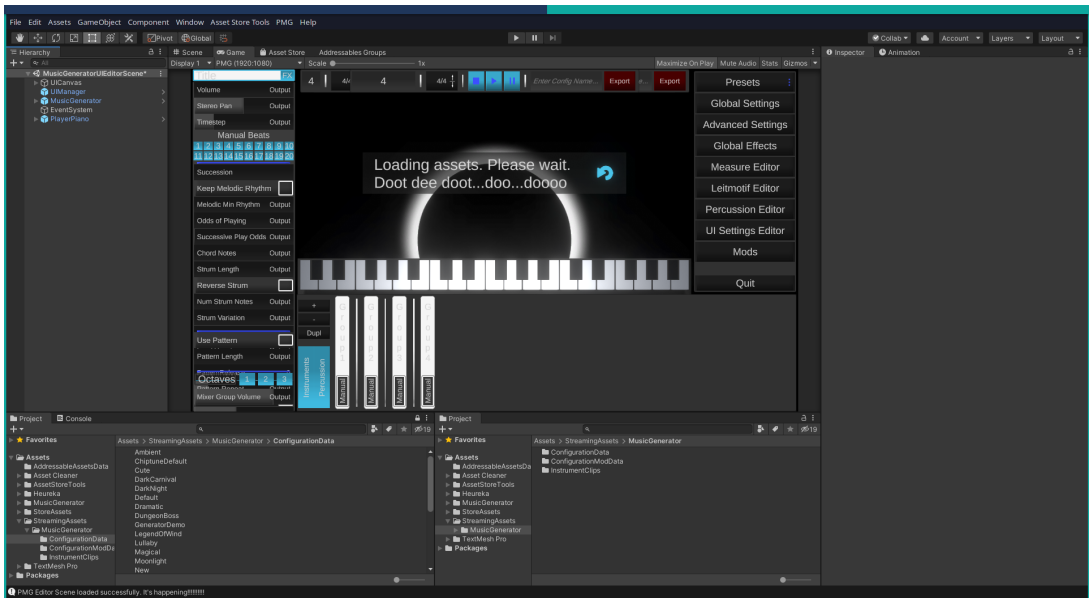
Index

Setup top^

1. Save Any open scenes and work.
2. Import the Procedural Music Generator asset into your project. Please import all assets.
3. Run the following Unity Editor menu commands. **It is important to agree to install any package dependencies requested.**
   a. UnityMenu>PMG>First Time Setup>1-Scene Setup (make sure you import TMP Essentials. TMP extras is not necessary)
   b. UnityMenu>PMG>First Time Setup>2-Addressable Setup



From this scene you can run the UI editor to create new music configurations to play in your project.



*Note: For most systems, the performance in the editor should be fine to create configurations and not require making a standalone build. **Please ensure you're not profiling the scene as it carries a lot of overhead.** You can also disable many of the UI's FX in the UI Settings Editor to further improve performance. If you're still not running at an acceptable framerate within the Unity Editor, simply build the scene here into a standalone executable to run. It's only required for generating the configurations. The generator itself has a very small cpu footprint in your project when playing configurations otherwise.

Discord top^

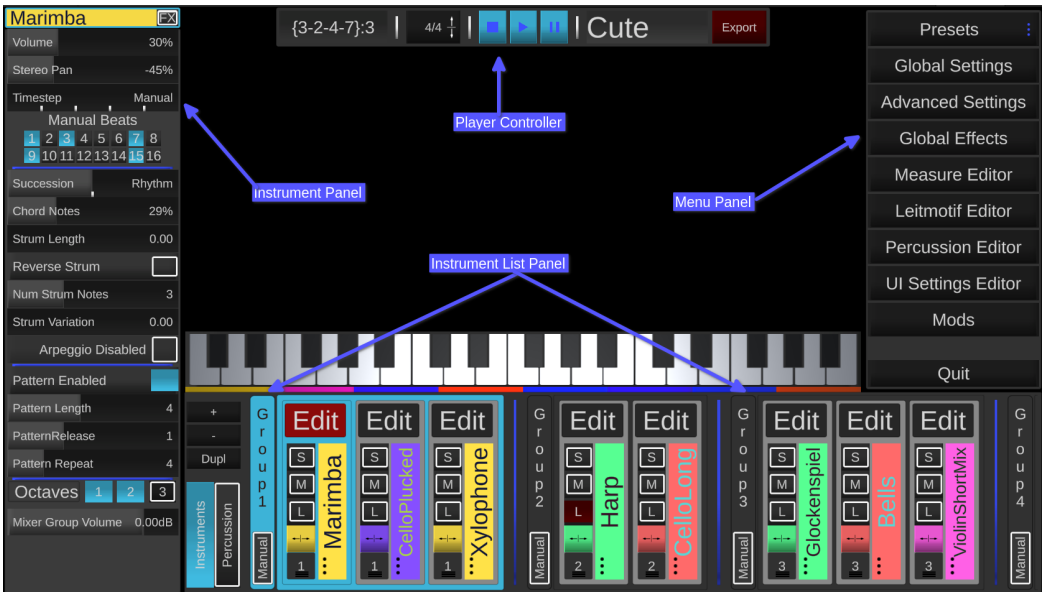
Please join our discord for additional tutorials, advice, chat, configuration sharing, update info, bug reporting etc: https://discord.gg/wsdmhS7VvR

Creating Configurations top^

General Workflow top^

The general workflow for the procedural music generator is to create configurations using the UI Editor scene, then loading and playing those configurations in the background of your app (where it will still be available to adjust during runtime as needed). Please see the included demo scene for an example implementation.

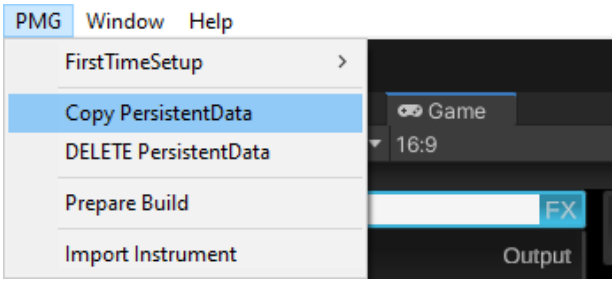
- Run the scene to start the UI Editor. The Default configuration will automatically load.



- You can hold Left Shift over most UI Elements for a tooltip regarding the purpose of that element.
- You can edit any preset configuration and export the changes under a new configuration name using the export button. **Changes are *only* saved when the export**

button is pressed and configurations are *only* overwritten if a configuration is saved to an existing configuration name. All exported configurations will then appear under the preset dropdown unless otherwise deleted.

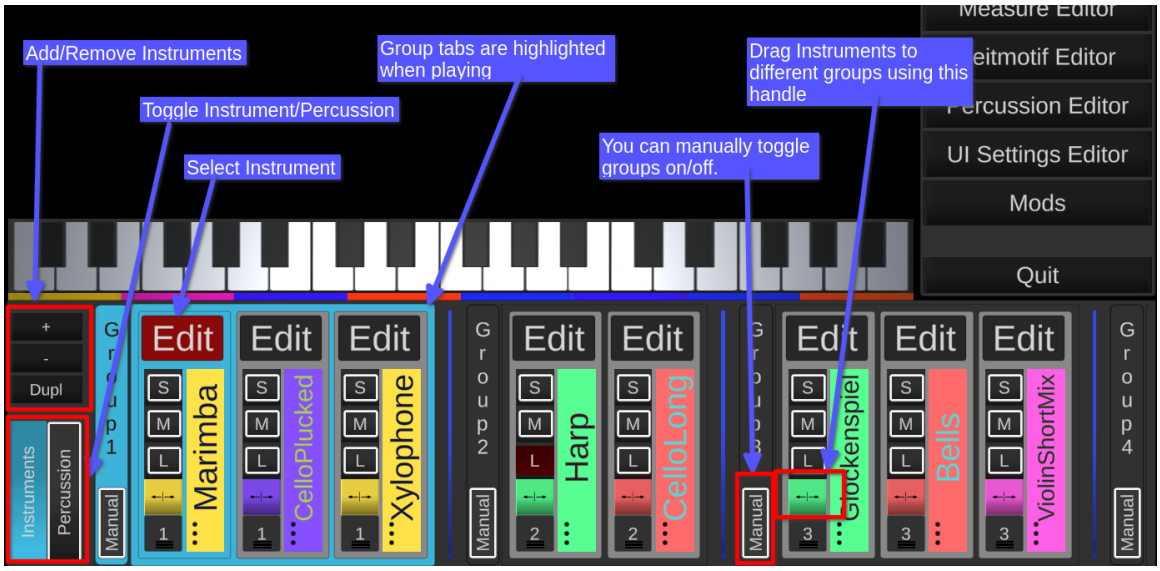
- You can also select the 'New' preset option for a blank configuration.
- Configurations are initially exported to the persistent data directory for your platform. See: https://docs.unity3d.com/ScriptReference/Application-persistentDataPath.html
- To include the configuration in your built project you will need to copy the configurations to your StreamingAssets directory. There is a menu command under PMG>CopyPersistentData to streamline this. Note: this will override any existing configurations in your project if they share the same name.



- You can likewise delete your *persistent* data configurations using the subsequent command. This is permanent. The configurations already copied to your streaming assets need to be deleted manually (this is by design, It really is permanent, and entirely on you if you lose work at this point :P).
- **Please note that loading persistent data versions of configurations are prioritized when loading configurations in the editor although they will be unavailable in your built application** until copied over (unless that application happens to be running on your development machine).

Instrument List Panel top^

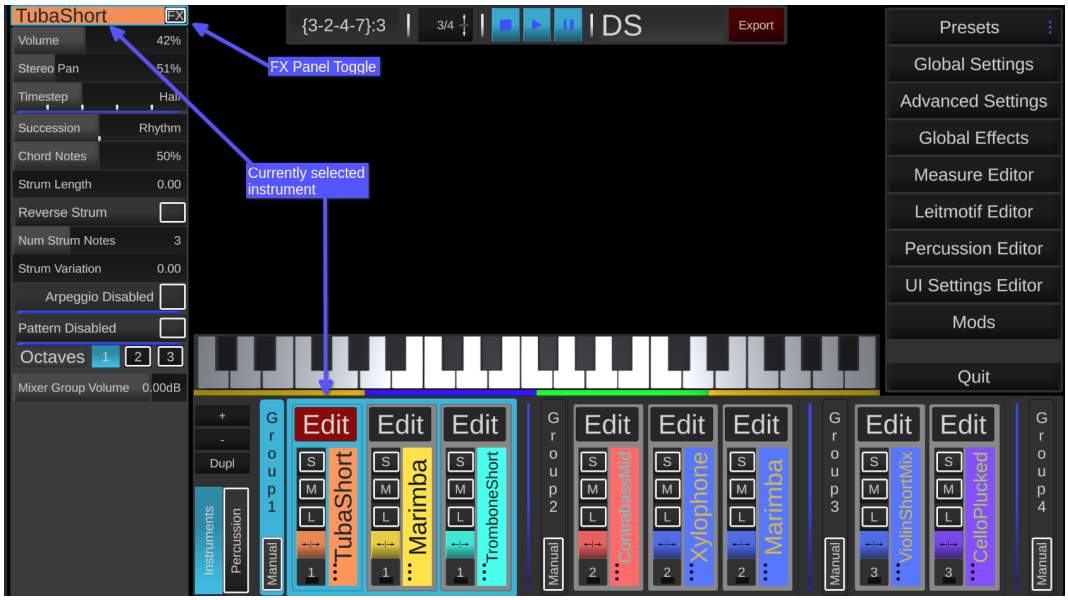
The Instrument List panel will show the list of instruments and percussion for the currently loaded configuration.



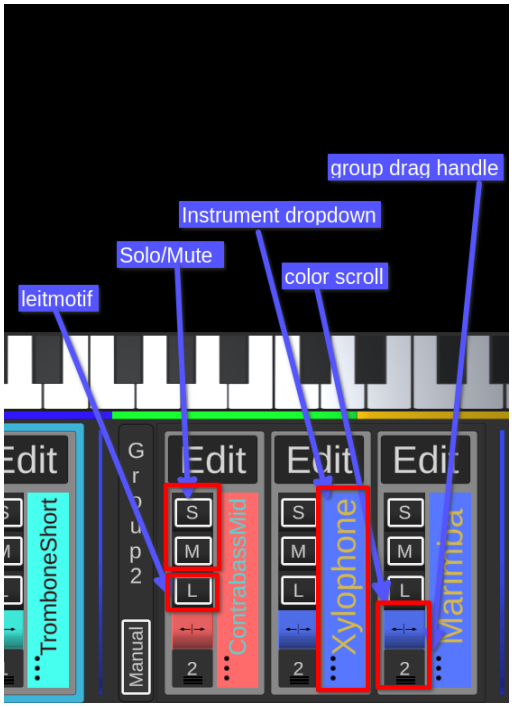
- You can drag/scroll the entire list of instruments if they overflow the area at the bottom.
- Adding an instrument will create a new instrument object placed at the end of the first group.
- The remove instrument button will remove the currently selected instrument (this can't be undone).
- You can drag an instrument from one group to another using the drag handle on the instrument object.
- Groups are a selection of instruments that either play/don't play together to create dynamics during play. Their odds can be set in the General Settings panel and their behavior (whether groups are chosen linearly to create a sense of crescendo or randomly) can be set in the Advanced Settings panel.
- While the groups are chosen programmatically to play (they are highlighted when generating notes), you can toggle groups on/off for the duration of the play session using the manual override toggle (this is not saved and only used to isolate groups of instruments for easier editing) the group odds will resume normally when all manual group toggles are turned off. Note that if any manual toggle is on, we're effectively in 'manual group' mode and only manually enabled groups will play.

Instrument Panel top^

The Instrument Panel on the left side of the screen will show the settings for the selected instrument. **Hold Left Shift over ui elements for additional informational tooltips and links.**



Instrument Object top^

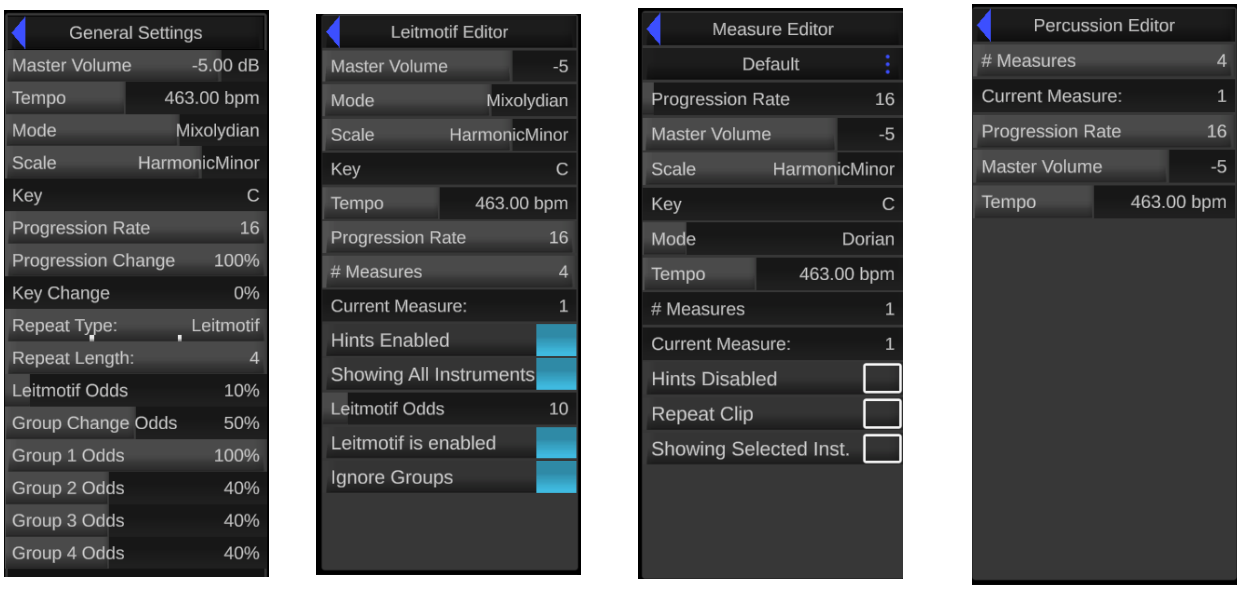


- Instrument objects do not need to be selected to use these ui elements (as opposed to those in the instrument panel on the left side).
- Solo will mute all other instruments. Toggle again to re-enable all instruments. Un-muting another instrument will un-solo the instrument while leaving the others muted.
- Mouse scroll over the color scroller to change the note colors of the instrument. ● Drag instruments from one group to another to change its play group.

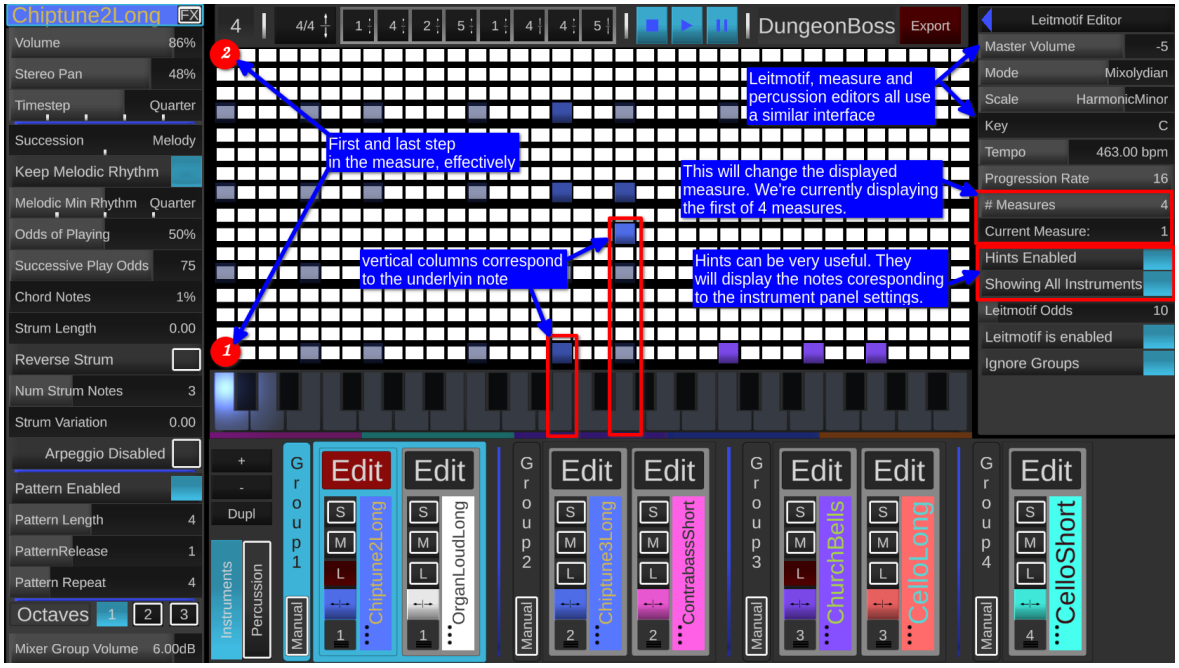
Editor Displays top^

The Leitmotif Editor, Measure Editor, and Percussion Editor will all use very similar interfaces. See documentation for each for any differences between them. Please note, that with the

exception of the measure editor, the values between all of these panels refer to the same configuration values (increasing the tempo on any of them applies to the entire configuration. It's simply exposed on the panels for convenience.

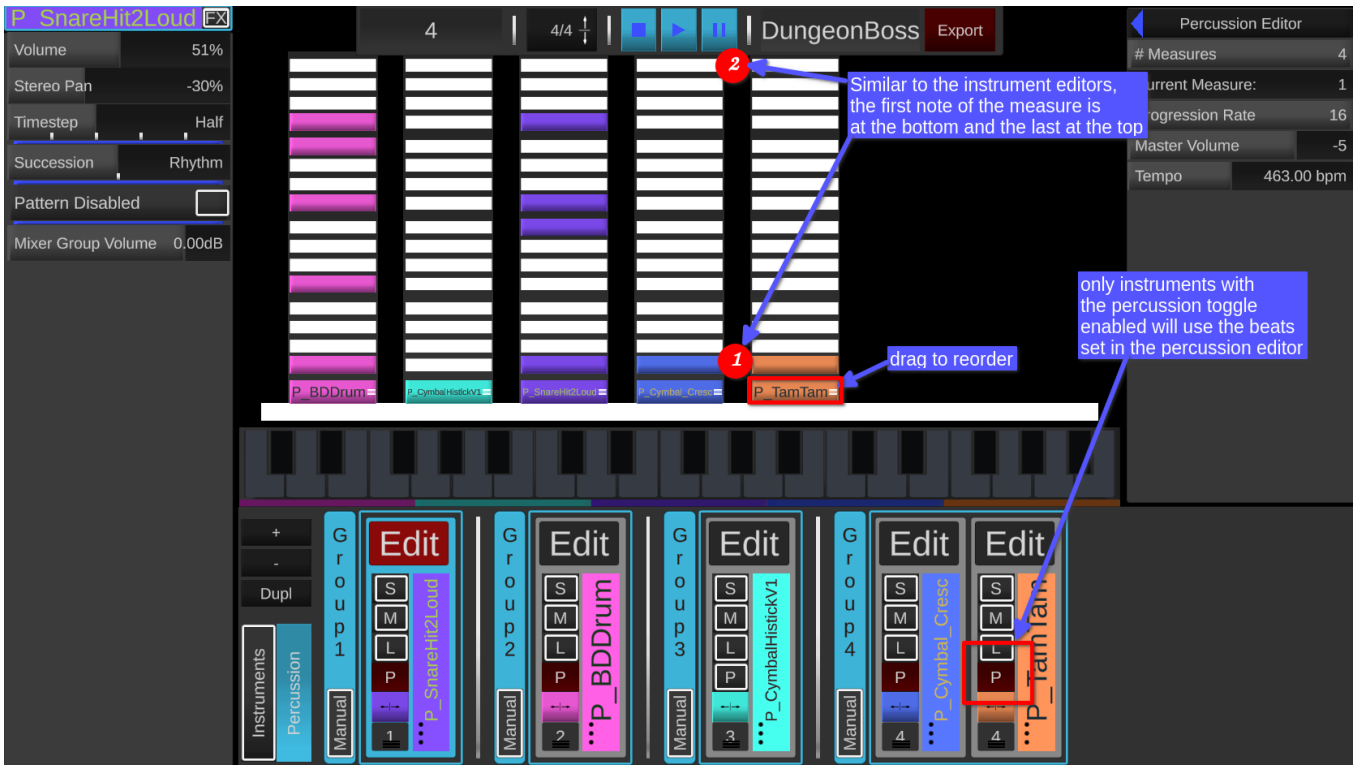


The general flow for all of the editor displays is to click on the notes you want played, click again to remove. A highlight will show on the keyboard for the selected note, and when a note is clicked, all notes for that time step of the measure will play (to assist hearing what it will sound like for chords).



*Click notes to add, click again to remove.

- The Chord Progression Sliders at the top (to the left of the stop/play buttons) controls the chord progression for the display editors. While selected notes do not need to correspond to this, the shown hints will be guided by the chosen progression. Scroll the mouse button over each to change.
- The Showing Selected Inst/Show All Instruments toggle will show the added notes for all instruments (though only the selected instrument will have its notes added/removed.
- The Hints toggle will show/hide hints for that instrument based on the Chord Progression (at the top)



The percussion display for the editors is slightly different, with all instruments showing in an isolated editor. You can drag the bottom icon to rearrange, or use the slider at the bottom to scroll the entirety of the display if there are more instruments than will fit.

Leitmotif Editor top^

The leitmotif editor allows the user to create a refrain or theme with an explicit set of notes that the rest of the generator can build around. Instruments with their Leitmotif toggle will only play the explicitly set notes when the leitmotif is playing (determined at the beginning of

a measure based on the Leitmotif Odds setting).

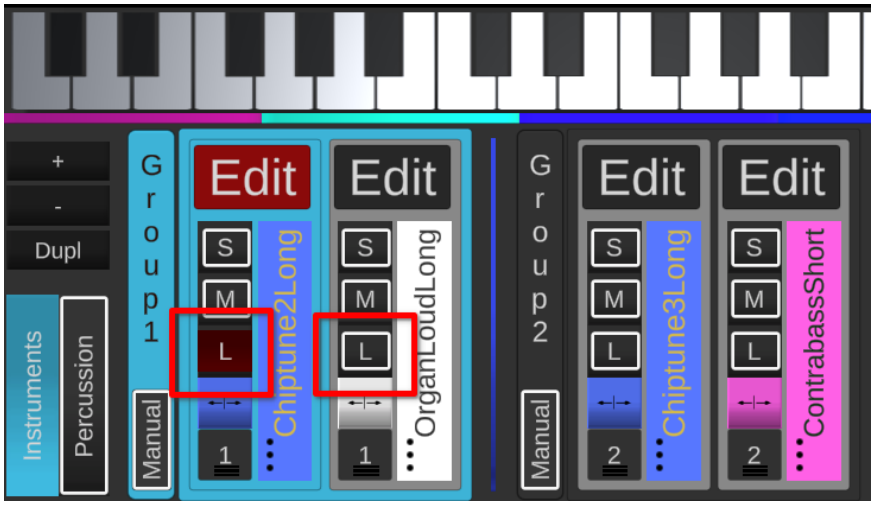
To use a leitmotif: Correspond

- Ensure the leitmotif feature is enabled. You can do so in two places:
    1. Within the Global Settings
    2. Within the Leitmotif Editor

*The General Settings, Leitmotif Editor UI Menu Panels respectively. As general advice: the leitmotif odds should probably either have *really low odds* (even 10 here is a little high) , or have a very slight touch (the suggestion of a melody or something). Whilst it's easy to feel great about your leitmotif you made, on the 1003238473 playthrough it will get a little grating. Trust me on this.

Only instruments with their leitmotif toggled on will play their explicitly set notes within the editor. The others will generate notes normally, adjusting to the chord progression of the leitmotif.

Please note that if playing within the leitmotif editor that only the leitmotif notes will play, while in the normal play, the non-leitmotif instruments will generate normally. This is to assist the setting of the leitmotif values by isolating only those notes and instruments.



*Here, the chiptune will use the leitmotif while the others will generate notes normally.

- Also note that the leitmotif for percussion will override any forced percussion settings on the instrument for the duration of the leitmotif if the instrument is toggled as a leitmotif instrument, otherwise it will respect the forced percussion settings normally.

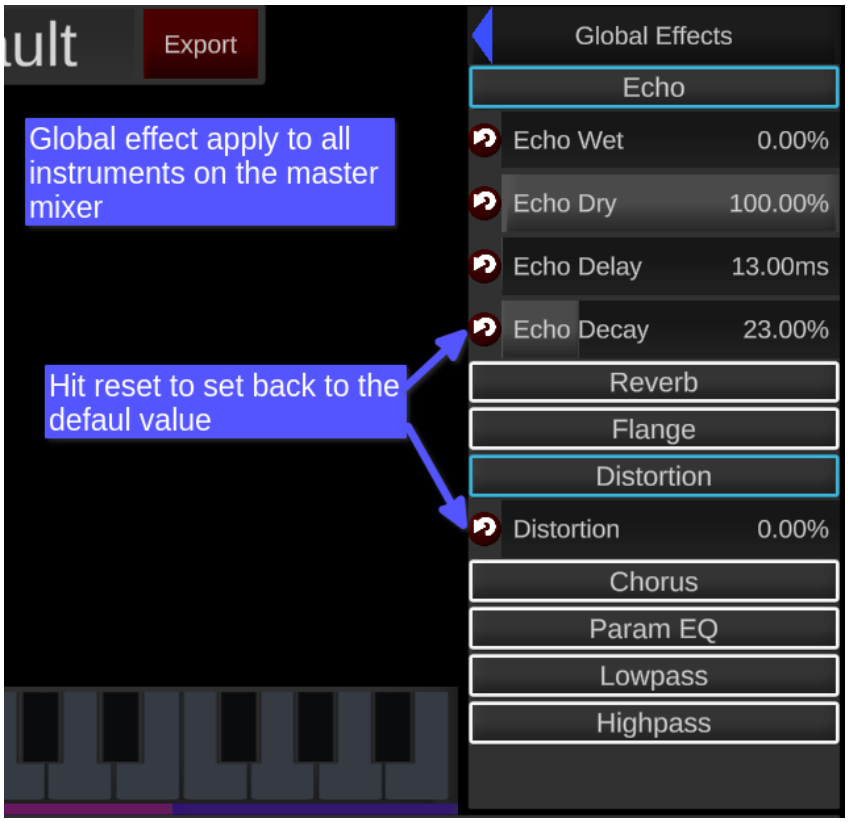
Measure Editor top^

The Measure Editor can be used to create one-off clips intended to be used as SFX type effects (opening a chest, leveling up, etc), and not as a replacement for general music configurations. There is no randomization here and notes are only played explicitly. This feature is mostly included to add an additional way to add audio effects that match the overall

theme/instruments for your project.

- Please see the example in the Demo Scene scripts for loading/playing of these configurations.
- Please note that the clip-configurations created by the measure editor are entirely separate from the general configuration and must be triggered manually. The exported configuration name is also independent and the configurations are cached in a separate folder.
- The editor display of the Measure editor is similar to the Leitmotif: click to add note of the currently selected instrument, click again to remove, etc.

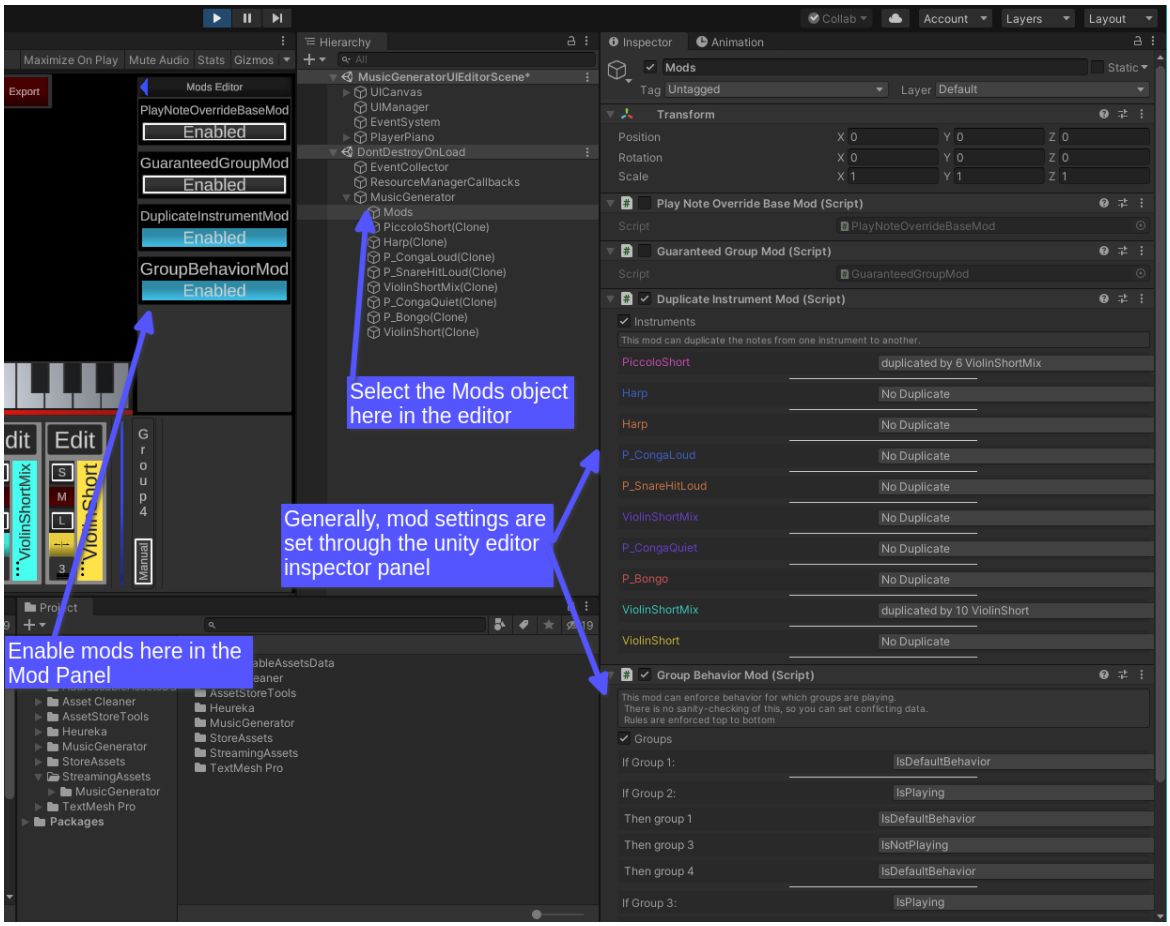
GlobalEffects top^



- Global effects apply the effects on the master mixer and apply to all instruments. ● Any instrument effects set in the instrument panel are mixed on the instrument channel (which is then mixed into the master channel). If you have both global reverb and instrument reverb, you'll get the instrument reverb mixed into the global reverb and probably have a lot of reverb :P.
- The reset button is advised if everything has just gone wrong.

Mods top^

You can easily extend or modify the generator behavior per configuration. A few existing mods are setup already (duplicate instrument, group behavior, etc). The data is save per configuration if the mod is enabled.

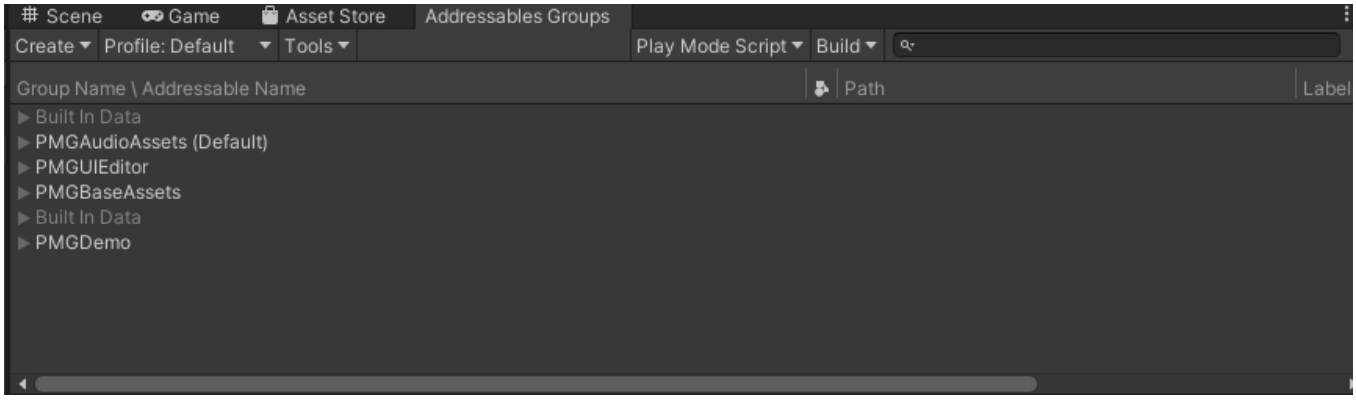


If you'd like to extend the generator behavior yourself, there is a base mod class located at MusicGenerator/Assets/Scripts/Mods/GeneratorMod. Simply override the base class and script in your behavior. See the example mod at MusicGenerator/Assets/Scripts/Mods/ExampleMod.

Packaging Notes top^

There are far too many use cases to try to cover here. The procedural music generator uses the addressable system. In the setup scripts, we build the addressable assets for your current platform for the editor (not the target platform). If you're targeting additional platforms, you'll need to generate the addressable assets manually for those platforms.

For many cases, simply using the Unity Editor build tools will auto generate your addressables. I've also included a build script that should build your target platform addressables locally under the Unity Menu>PMG>Prepare Build menu command:



*You're not likely to need the addressable assets for the demo or UI editor in your project and they've been separated out. However, the included build script will build them (the ui editor needs to build them).However, for all intents and purposes you only need the PMGAudioAssets addressables asset for your project.

- For others (remote bundles, remote build, etc) you'll need to build these manually or adjust settings accordingly. See the unity documentation for details: https://docs.unity3d.com/Packages/com.unity.addressables@1.8/manual/index.html ● We're currently using addressables 1.8.5. It's entirely likely I'll update that dependency someday and forget to update this documentation, so please see the relevant documentation if that's the case: https://docs.unity3d.com/Manual/com.unity.addressables.html
- The PMG settings file is located under the MusicGenerator/Assets/AddressableAssetsData if you need to update remote or target settings or change the group settings.

Custom Instruments top^

You are able to create custom instruments for the generator. It will require audio samples for 36 notes containing the notes for 3 octaves (from C to a C 2 octaves higher) (my genuine apologies, this is a pain in the a** :P See the section on 3rd Party Audio Assets below). Please ensure the samples have had any whitespace removed in the audio sample at the beginning. The audio attack should happen immediately in the samples (otherwise timing is off in the player as the instrument sounds like it's lagging behind).

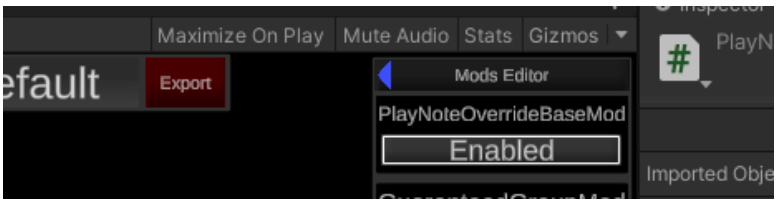
In order to incorporate the instrument, place a directory at:
MusicGenerator/Assets/Audio/YourInstrumentName

Place the audio samples in this directory. They must be named 1-36 starting at low-C through high-C 2 octaves higher. See the other instruments for an example.

Open the tab located at UnityEditorMenu>PMG>Import Instrument.
Drag your directory to the "Instrument Directory" field and name your instrument with the same name.
Click "Import Instrument" and agree to save to save (if you're adding multiple instruments, feel free to select "I have more to import" and repeat the steps above before finalizing things.)

3rd Party Audio Assets top^

I've added a mod to the generator that will aid in routing the generator notes through whatever 3rd party audio setup you have, whether it's midi, vst, or some other setup. Essentially you'll just be forwarding the data externally to whatever setup you have.
You'll need to enable the "PlayNoteOverrideBaseMod."



This will essentially grab the data for the PlayNote event of the generator and telling it to suppress the playing of it. You can then forward the data however you need.

You will need to script in the behavior for this. Please see PlayNoteOverrideBaseMod::OnNotePlayed for where this data is surfaced to you.:

```
private bool OnNotePlayed( object source, NotePlayedArgs args )
{
    // Do something with args here:
    var noteIndex:int = args.Note; //< so, 0='C' 1='C#' 2 = 'D' etc.
    var instrumentIndex:int = args.InstrumentIndex; //< you'll have to reference your setup to know which instrument this refers to
    var volume:float = args.Volume;
    var set = args.InstrumentSet; //< this has almost _all_ the other data.
    var instrument = set.Instruments[args.InstrumentIndex]; //< some instrument data
    var instrumentData = instrument.InstrumentData; //< all the relevant instrument data

    // Send whatever data you want to your other library:

    /* If you're trying to do everything in a single method, I'd just feed in this data to a data container with the relevant info, and in a LateUpdate
     process the data (i'm uncertain which data you need for your use case, so can't offer much pseudo code)
     you may want/need to add a new unityEvent in MusicGenerator::UpdateState() so you know each 'tick'. or better yet, add an event in
     MusicGenerator::Measure that is invoked upon 'TakeStep' (all the notes will be played during that step)
     Unfortunately, not everything is played in a single chunk to just override and get all the note data at once.*/

    // You can still play the note here by returning true

    return false; //< return false in order to keep the generator from playing the note. If, say you're having an external library play it
}
```