Guitar Unity Readme

Overview

Hi and thank you for purchasing this “Guitar based reaction game” starter kit ;-)

In this document I will give you a quick tutorial on how to install and use this package for your own project. If you have any further questions feel free to e-mail me at oliver@eberlei.de

Have fun.

Installing Guitar Unity

Create a new project in Unity, then, in the menu bar, click on Assets -> Import Package -> Custom Package and select the GuitarUnity.unitypackage file you just downloaded. Click “Import” to import all the assets in the package.

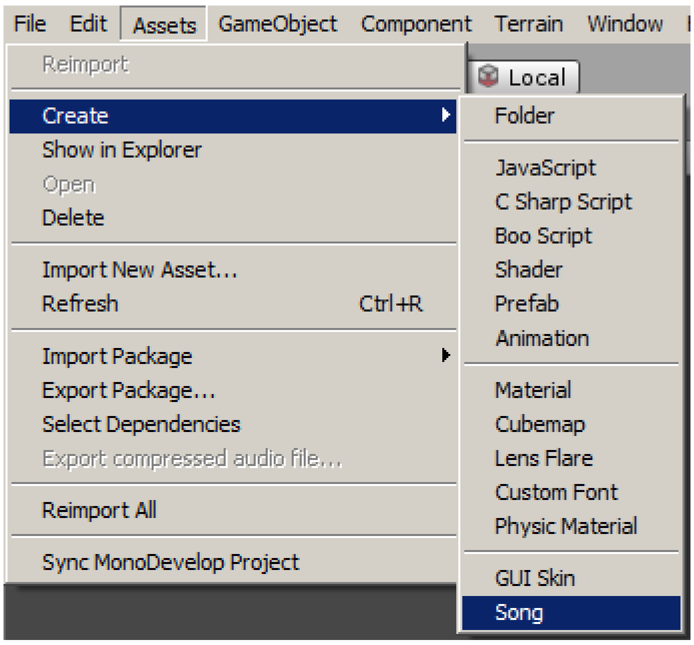
Using Guitar Unity

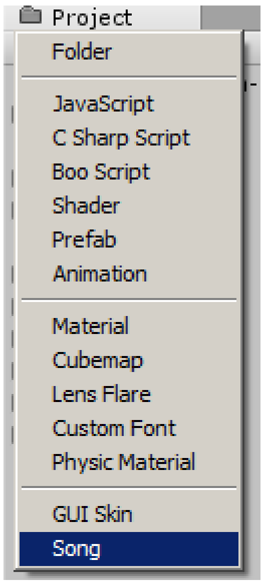
First of all, you have to make sure the GuitarUnity scene is loaded. Just locate it in the Project view and double click it. This is important even if you just edit the song files in the song editor. Otherwise the scripts will not be able to find the guitar object and fail to execute.

For each song you want to create you have to follow these three easy steps:

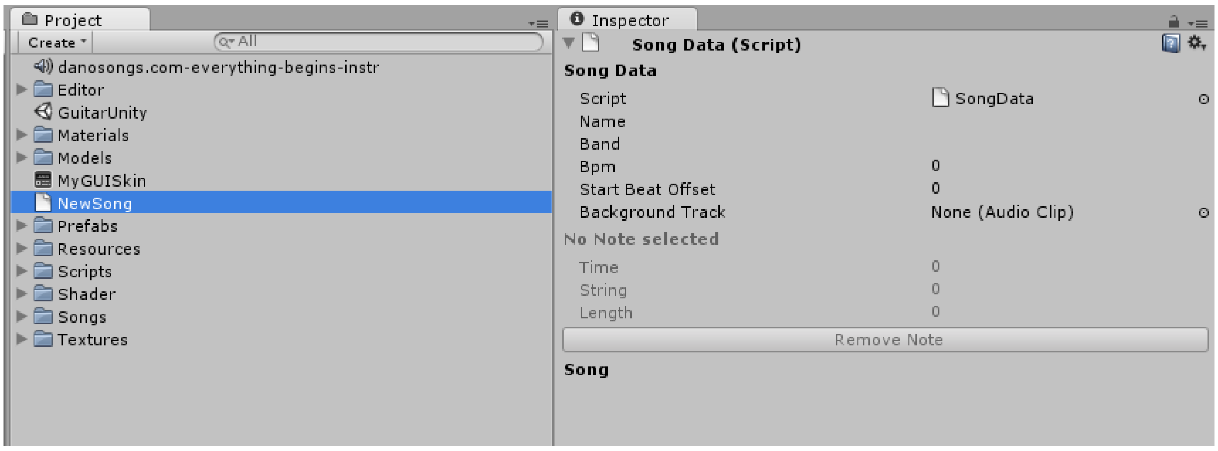
1. How to create a song

Either click Assets -> Create -> Song in the navigation menu or click Create -> Song in the Project View. A new Asset named NewSong will appear in your project view.





2. How to edit a song

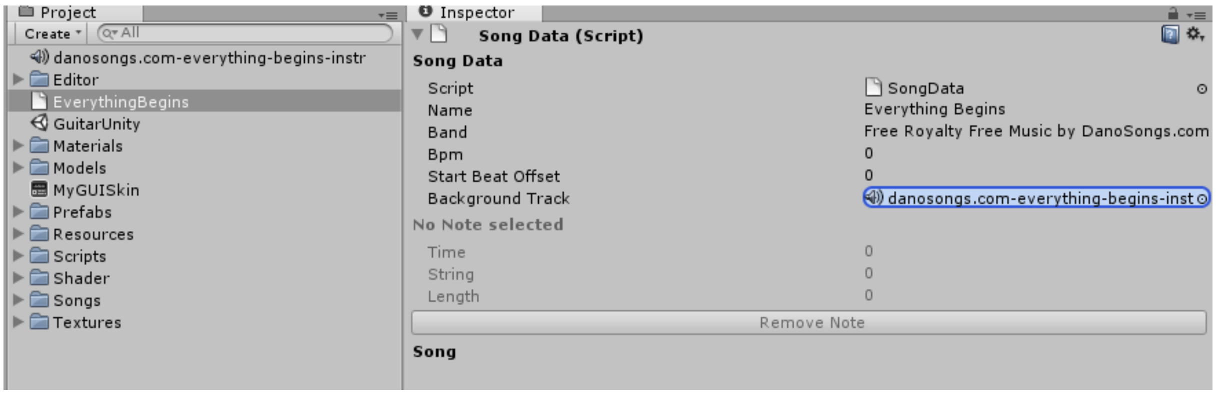


```
Project                                   Inspector
Create ▾   All                            Song Data (Script)
danosongs.com-everything-begins-instr     Song Data
Editor                                      Script              SongData
GuitarUnity                                 Name
Materials                                   Band
Models                                      Bpm                 0
MyGUISkin                                   Start Beat Offset   0
NewSong                                     Background Track    None (Audio Clip)
Prefabs                                   No Note selected
Resources                                   Time                0
Scripts                                     String              0
Shader                                      Length              0
Songs                                                Remove Note
Textures                                  Song
```

Select the song you want to edit in your project view. The song editor will show in the inspector.

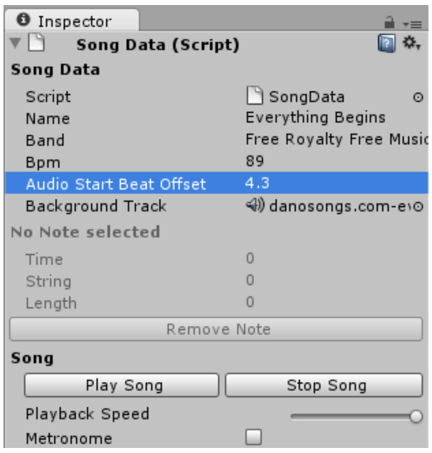
First, type in the name of the song and the band that created it. These values will be used in the main menu to describe the song.

If you haven't done so already, copy the sound file of the song you want to include in the game into the /Assets/ directory in your Guitar Unity project folder. Next you have to drag the file from the project window onto the Background Track element in the inspector.



```
Project                                   Inspector
Create ▾   All                            Song Data (Script)
danosongs.com-everything-begins-instr     Song Data
Editor                                      Script              SongData
EverythingBegins                            Name                Everything Begins
GuitarUnity                                 Band                Free Royalty Free Music by DanoSongs.com
Materials                                   Bpm                 0
Models                                      Start Beat Offset   0
MyGUISkin                                   Background Track    danosongs.com-everything-begins-inst
Prefabs                                   No Note selected
Resources                                   Time                0
Scripts                                     String              0
Shader                                      Length              0
Songs                                                Remove Note
Textures                                  Song
```

If you've done everything correctly, you should hear the song playing when pressing "Play Song". You will notice that the editor window does not move yet, this is because we have not set the songs Beats per Minute.



```
Inspector
Song Data (Script)
Song Data
  Script                    SongData
  Name                      Everything Begins
  Band                      Free Royalty Free Music
  Bpm                       89
  Audio Start Beat Offset   4.3
  Background Track          danosongs.com-e
No Note selected
  Time                      0
  String                    0
  Length                    0
             Remove Note
Song
  Play Song        Stop Song
  Playback Speed
  Metronome
```

The Beats per Minute (Bpm) are necessary to align the speed of the song with the speed the notes move during the game. It is essential that this value is correct; otherwise it will be hard for the player to get the correct rhythm. If you don't know the Bpm of the song, you can enable the Metronome and hit play. Now try to align the metronome beeps to the beat.

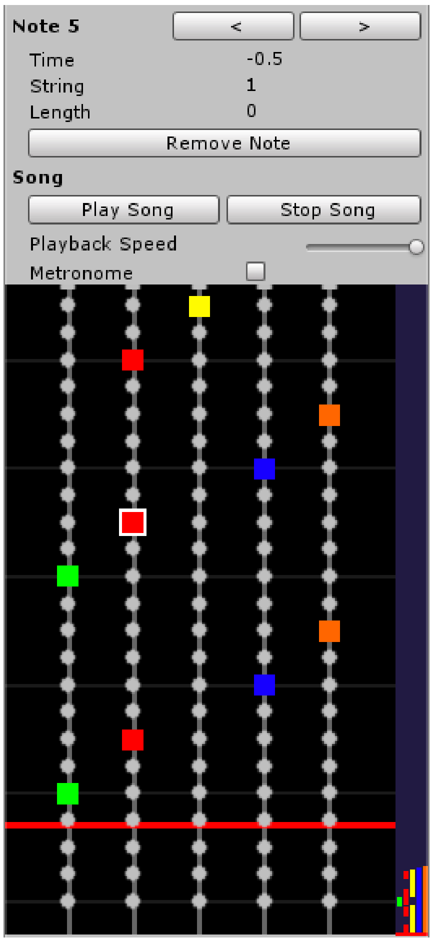
Hint: In my opinion it is easier to get the right Bpm if you have more Metronome Beeps. Try doubling the Bpm, which results in twice as many beeps.

Sometimes you have the right speed, but all the beats are still offset by a certain amount of time. This is where the “Audio Start Beat Offset” comes in. This value delays the playback of the song so the Metronome beep actually occurs at the beginning of a beat.

Notice: This value is not measured in seconds, but in beats. For example 0.5 means the song is played half a beat later. If you would change your Bpm later, the time offset changes with it. I decided to measure everything in beats so it is easier to offset the song by whole beats. Also, you don’t have to redo the whole song if you have to slightly change the Bpm in the end.

In the editor, this value only offsets the beats by the fraction you provide. Whole beats are ignored. But during the game, the song is also delayed by the whole beats. This is helpful to time the beginning of the song with the 4-3-2-1-Go Countdown.

You can adjust the Playback Speed slider to slow the song down. This might help you to align the Metronome Beep precisely. The playback speed does not affect the speed of the song during the game. It’s just a tool to help you edit your song.



Time -0.5
String 1
Length 0

Once you are happy with the timing of the song, you can start adding notes.

You can add notes by clicking on the circles in the editor with your left mouse button. If you click on an existing note, it is selected. And right clicking an existing note deletes it.

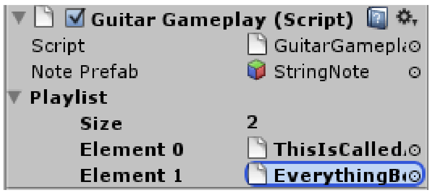
When a note is selected, its values are shown above the editor in the inspector, and you can fine tune its time (measured in beats), string position and length. If the length is bigger than zero, a trail is added to the note.

There are also some keyboard shortcuts to help you speed up your editing process:

Left Arrow	Select Previous Note
Right Arrow	Select Next Note
Up Arrow	Advance song by one beat
Down Arrow	Reverse song by one beat
DEL	Delete currently selected note

To quickly navigate to a specific part of the song, you can click on the overview of your song on the right. (The one with the dark blue, purple-ish background)

3. Add the song to the playlist



Script GuitarGamepl
Note Prefab StringNote
Playlist
Size 2
Element 0 ThisIsCalled.
Element 1 EverythingB

Finally, select the Guitar GameObject in the scene. In the Inspector you can see the Playlist element in the Guitar Gameplay component.

Drag your newly created song from the Project View into this playlist to add it to the list of songs that can be played in the game.

Customizing the visuals

The shader that is used for the notes and buttons uses the alphachannel to include the color changes for the different strings…

Class descriptions

Editor/SongDataEditor.cs

This is one of the two major components in my project, the song editor.
It handles the creation of Songs (via Assets -> Create -> Song) and draws the editor when a song is selected in the Inspector.

Editor/StringButtonEditor.cs

This is the editor that is shown in the inspector when you select one of the five note buttons in the scene. It basically just draws the default inspector, with the addition of a light intensity slider which controls the light the button emits while it is pressed.

This editor also makes sure that the color changes are applied to the in game objects immediately.

Menus/BaseMenu.cs

All menus that are used in the game inherit from this class.

Menus/EndofSongMenu.cs

This is the menu that is shown after a song has finished playing. It lists the score of the player and some statistics on how well he did.

Menus/InGameMenu.cs

This menu is shown when the user hits escape during a song.

Menus/MainMenu.cs

This menu is shown in the beginning. It lists the songs the user can select.

GuitarGameplay.cs

This is the second major component to this project. It handles all the gameplay mechanics.

KeyboardControl.cs

This class processes the users input.

MyMath.cs

This is a static class which provides two helper functions, which are used throughout the project. Because the class and the functions are labeled static, they are available in all the other scripts.

SongData.cs

This class provides the internal data structure for the songs. This class is also serialized and deserialized to save the data.

SongPlayer.cs

This class handles the playback of the songs.

StringButton.cs
This class does not do anything. It just provides two variables which can be edited in the inspector. The StringButtonEditor then uses these two variables to apply the values where necessary.

Timer.cs
A small helper class I used in the editor to measure its performance.