

Using Cantabile and the MuLab Plugin Together

Derek Cook © 2022



Introduction and Background

This guide shows you how to setup and configure MuTools' [MuLab](#) VST plugin within Top Ten Software's [Cantabile](#) live host to create a live looping environment within Cantabile.

I was looking for a solution where I could perform live looping within Cantabile without needing to link to external software like Ableton Live, as I wanted everything working inside Cantabile songs.

After evaluating a few different looping applications (including Ableton Live) I settled on the [MuLab](#) VST plugin, as it appears to have a huge amount of flexibility. This means it also has quite a steep learning curve, which is why I wrote this guide; not only to help other Cantabile Users but to document how I set it up so that I have something to refer to in a year or so from now, when what I did has escaped my memory!

I hope that you as an interested Cantabile user will also find this guide useful.

This guide was written using Cantabile Performer 4 (latest version), and you may need to adapt the approaches given here for other versions of Cantabile.

The example assumes that you have already installed the MuLab plugin on your system and that Cantabile has detected it and it is available for use.

The example is written very much around how I use Cantabile and my looping examples that will be my starter projects. However, you will hopefully be able follow what is being done, and adapt the principles to your own requirements and ways of using Cantabile.

Document Contents

We'll be covering the following in this document.

| | |
|---|----|
| Looper Design Approach | 2 |
| Creating the MuLab Scenes in your Favourite DAW | 3 |
| Creating the Cantabile Linked Rack for MuLab | 9 |
| Creating the MuLab Project | 11 |
| Creating the MuLab Audio Tracks | 29 |
| Setting up the Cantabile Linked Rack for MuLab | 35 |
| Selecting MuLab Scenes from your Keyboard..... | 42 |
| Selecting MuLab Scenes from an External Controller | 43 |
| Showing MuLab Scene Changes in the Cantabile Window | 46 |
| Creating Linked Rack States to select Different MuLab Projects..... | 48 |
| Conclusion | 54 |

Looper Design Approach

I decided to take the approach of creating a linked rack that hosts the MuLab plugin, with Rack States then used to store different MuLab Projects (the equivalent of Cantabile Songs).

Rack States can then be used within songs in the set list to select different MuLab Plugin Snapshots, where each Snapshot can select a different MuLab Project.

Whilst you could use a single linked rack across many set lists, my intent is to have a linked rack for each set list that requires it, as I like to keep things organised set list by setlist, but this is very much personal choice.

The linked rack also has all the required bindings and routings encapsulated within it, which makes it easy to reuse.

In Cantabile **Songs**, you can simply drop in the MuLab **Linked Rack** that I intend to be a unique copy for each setlist, and in each Cantabile **Song** where you want a MuLab **Project** running, you simply need to select the Linked Rack State that sets the required plugin snapshot/MuLab project.

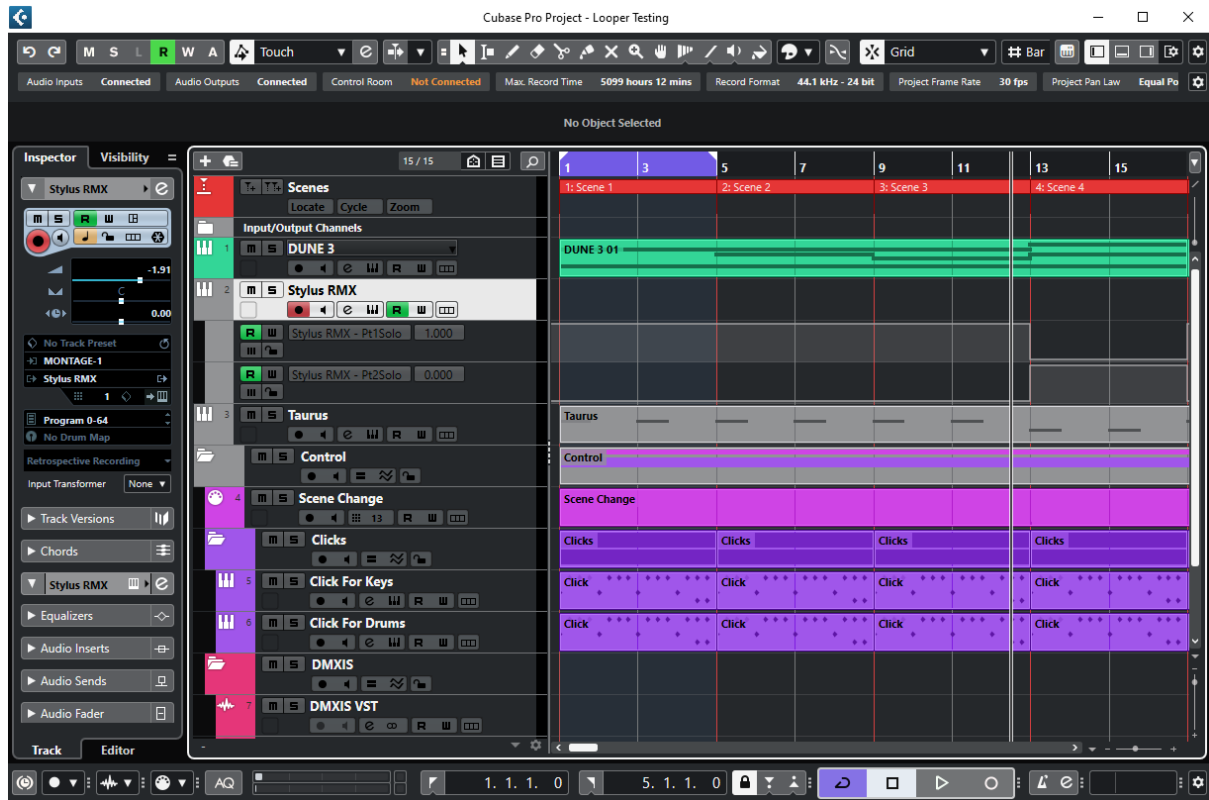
I also wanted a mechanism to show which MuLab scene is running, and I have used Custom Buttons on the Cantabile **Controller Bar** to achieve that, which I will also describe in this guide. I intend to use the Scene Buttons on my Yamaha Montage 7 to select the Scenes within a MuLab project, and as I will be setting up the scenes to only change on Bar boundaries, I wanted a means of indicating in Cantabile when a scene selection commanded on my Montage 7 (which I can make at any time) becomes active within MuLab.

With all that in mind, let's dive in, as we will need to go through quite a few steps before we get there. In outline we need to do the following:

- Create the Scenes that will be loaded into MuLab in your Favourite DAW;
- Create the Cantabile Linked Rack containing MuLab;
- Create the MuLab Project;
- Create the MuLab Audio Tracks;
- Set up the Cantabile Linked Rack for MuLab;
- Setup how to select MuLab Scenes from your Keyboard;
- Setup how to select MuLab Scenes from an External Controller;
- Setup how to show MuLab Scene Changes in the Cantabile Window;
- Setup Linked Rack States to select Different MuLab Projects.

Creating the MuLab Scenes in your Favourite DAW

I guess you could also use MuLab to do this step as well, but Cubase Pro is my preferred DAW of choice within which I am creating the source songs, and I only intend to use MuLab as a looper in Cantabile. Therefore, I created my Scenes in Cubase. Below is my example project in Cubase, and I wanted to test looping both MIDI and Audio data.



This Cubase project is set out as follows:

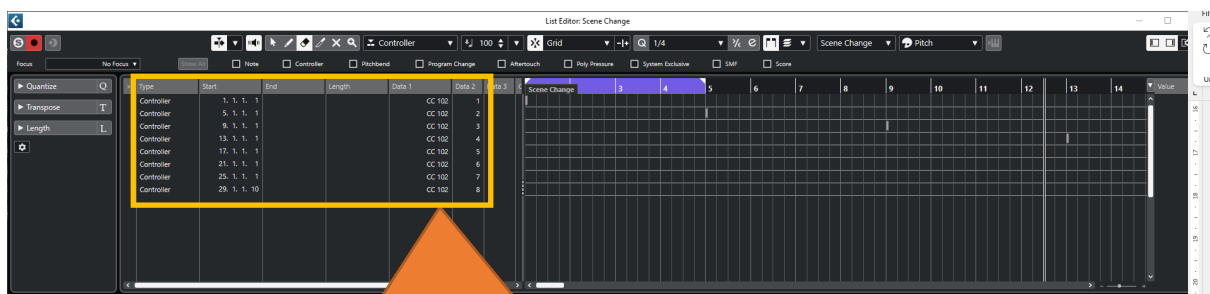
- **Scenes** is a Cubase *Marker Channel*, where I have created *Cycle Markers* for each Scene.
 - In the Cubase Timeline you can see I have the *Play Range* on the Scene 1 *Cycle Marker*, and the magenta colour of the *Play Range* (which is usually grey) indicates that Cubase is looping within the *Play Range*.
 - If you double click on a Scene *Cycle Marker*, then the play range is set to match that scene, so this is a quick way of checking scenes in Cubase and the cycle markers are crucial during the export process to get the data I need for MuLab sliced up into clips for the scenes.
- **Dune 3** is an instrument track for the amazing Dune 3 VST, which I am using to create an arpeggio for this example. For this track I wish to export MIDI, as in my example, I will use MIDI Clips within the scene to trigger Dune 3 in a Cantabile Song, which means I can still interact with Dune 3 via my controller surfaces.
- **Stylus RMX** is an instrument track for the Spectrasonics Stylus RMX Groove Module that I am using for some additional rhythm. I will export this track as audio.
 - I am new to Stylus and not sure if this is the best way to do this, but I have two parts for different scenes, with Part 1 unmuted in Scenes 1, 2 and 3, and Part 2 unmuted in Scene 4.

- **Taurus** is an instrument track for the Cherry Audio “Lowdown” Moog Taurus emulation to provide a little bit of bass. I will export this track as audio.
- **Control** is my folder for non-audio parts
 - **Scene Change** is a MIDI Track sending CCs, etc. on CH13 that will provide *Scene* feedback in Cantabile. More on this later
 - **Clicks** are tracks that I render to audio to provide my standard click.
 - It’s a bit of overkill, but I use Expansion BFD3 drum software to build my clicks.
 - I do this as I can build my own special cues. I have a scheme that uses certain sounds and pitches to let me know when things are happening, which you do not get from a basic metronome.
 - **DMXIS** is for my show lighting software (not used in this example), and it uses notes on MIDI channels 15 and 16 to control the DMXIS plugin to select banks and cues within a light show.
 - And also not shown used here, but I have CH14 reserved for controlling video if I am triggering that in a gig.

So basically, I need the MIDI and audio clips to be exported as follows:

- MIDI Clips for each Scene for:
 - Dune 3 (CH1);
 - Scene Change (CH13).¹
- Audio Clips for each Scene for:
 - Taurus;
 - Stylus RMX;
 - Clicks.

The Scene track/channel is best viewed in the Cubase List Editor to see what is happening, and is shown below.



MIDI CC Changes every four bars (my scenes are four bars long).

Basically, at the start each scene I have set a MIDI CC event. CC 102 is one of the unassigned CCs² for general use and I am using a value of 1 to represent Scene 1, a value of 2 to represent Scene 2, etc.

In Cantabile, this CC will be sent at the start of each scene and will be used to show that a commanded scene change (from the Scene Buttons on my Montage keyboard) has been activated within Cantabile when the scene starts playing. How this is achieved is described later on, but this track provides the source data for achieving it.

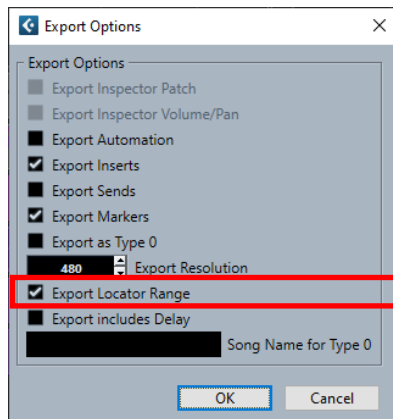
¹ If this was a complete show and not just an example, I might have Video Cues on CH14 and DMXIS Cues on CH15 and 16 in the same export.

² I.e. the MIDI Manufacturer’s Association (MMA) have not defined a purpose for the CC.

Exporting Cubase MIDI

Creating the MIDI exports is not as slick in Cubase as it is for audio data. What I have to do is:

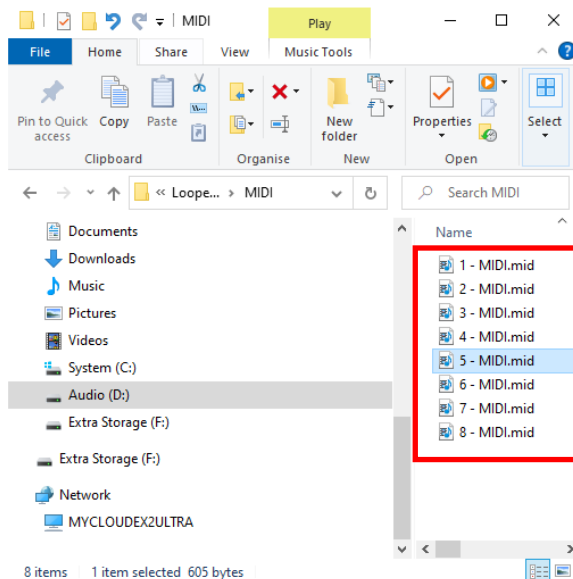
- Solo the Dune 3 and Scene Change Tracks (to avoid exporting other track data)
- Double click on the *Cycle Marker* for the Scene to export, and then
 - Select **File/Export/MIDI File...**
 - Type or select the file name
 - Select **Save**
 - Ensure the Export Options are set as follows (to only export the Locator Range)



This is the critical setting in Cubase to get the MIDI data between the locator Ranges. All other settings are the default export options.

- Click **OK** to export
- Repeat from the second bullet above for all of your scenes.

This gives me the following MIDI Clips that I will need to get into MuLab.



My exported MIDI files for each Scene.

Exporting the Audio Clips in Cubase is a little easier than it is for MIDI, as you can do it in one step for each track that you wish to export.

Select **File/Export/Audio Mixdown...**

And note the following comments on the export dialog.

Export Audio Mixdown

Channel Selection

Single Multiple

Search Channel

Output Channels

- Stereo Out
- Click

Instrument Tracks

- DUNE 3
- Stylus RMX
- Taurus
- Click For Keys
- Click For Drums
- Channels
- MXIS VST

File Location

Name: Stylus RMX

Path: D:\Cubase Projects\Test Area\Looper Testing\Mixdown

Preview: 1 - Stylus RMX.wav

Conflicts: Always Overwrite

File Format

Preset: No Preset

File Type: Wave

Sample Rate: 44.100 kHz

Bit Depth: 16 bit

Export as: Interleaved

3: "audio - 01 (L

Insert Broadcast Wave Chunk

Insert iXML Chunk

Insert Tempo Definition

Don't Use Wave Extensible Format

Don't Use RF64-Compliant File Format

Effects: Inserts and Strip

After Export: Do Nothing

Pool Folder: Enter Pool Folder Name

Export Range

Locators Cycle Markers

| ID | Name |
|-------------------------------------|-----------|
| <input checked="" type="checkbox"/> | 1 Scene 1 |
| <input checked="" type="checkbox"/> | 2 Scene 2 |
| <input checked="" type="checkbox"/> | 3 Scene 3 |

Export Queue

Keep Dialog Open

Update Display

Real Time Export

Use External MIDI Inputs

Export Audio

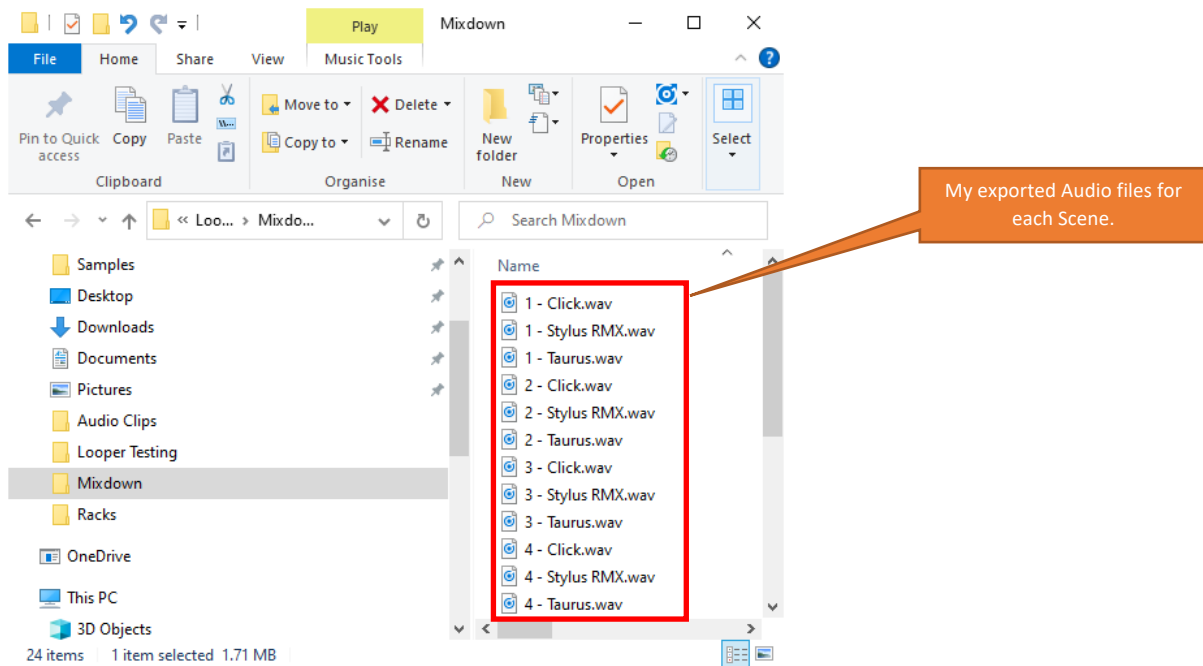
Select a suitable name for the export. Note how with Cycle Markers selected, the preview prepends the Cycle Marker ID so that each export is unique.

Select the Track that you wish to export (and repeat for each track of interest).

Select Cycle Markers and ensure that all cycle markers that you wish to export are ticked (scroll the list to see them all).

And click Export Audio for each track you wish to export.

Following this process, repeated for each track of interest, gives me the following audio tracks in the **Mixdown** folder.



Deciding Your File Structure

The first thing to do is to decide the structure of how you are going to store the information, noting that MuLab seems to rely on absolute file paths, not relative file paths like Cantabile.

From my main **Documents** Folder, my planned structure for including a MuLab Project within a Cantabile Set List and Song structure is as follows:

Documents Folder

 Cantabile Folder

 Setlist Folder

 Song Folder

 MuLab Folder

 MuLab Project File

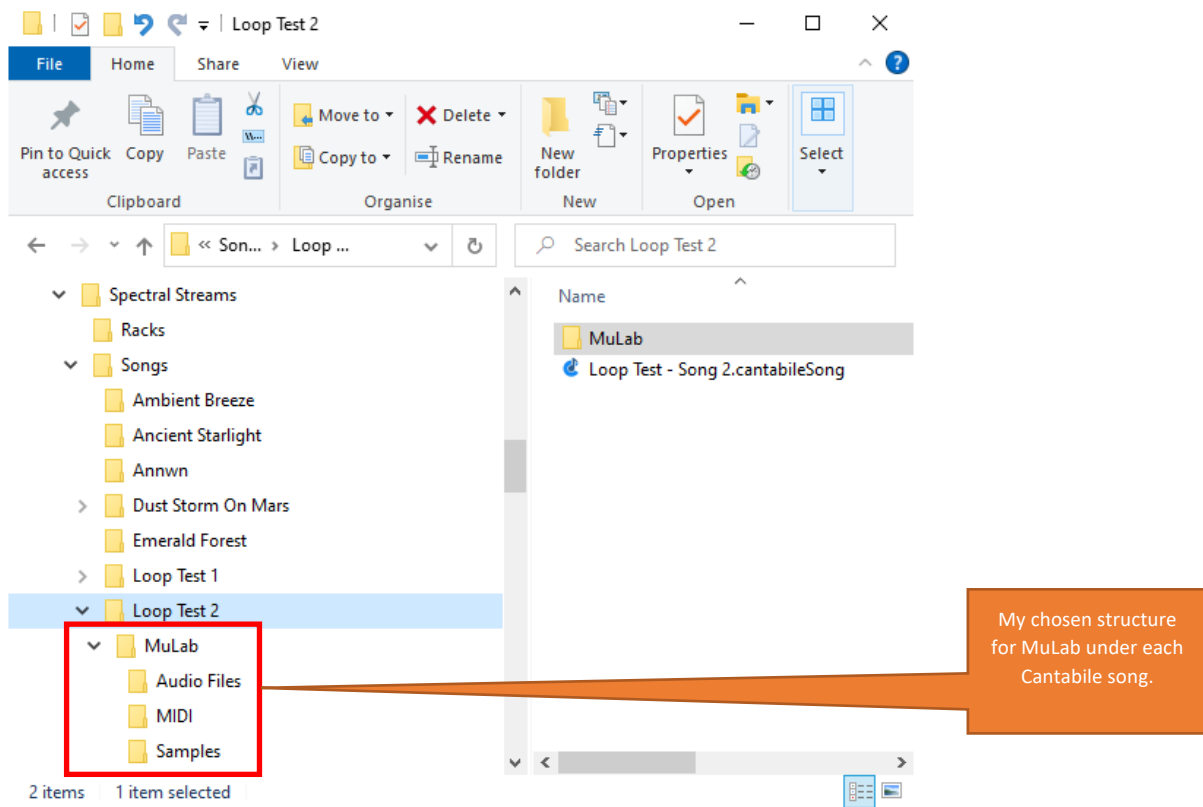
 Audio Files Folder

 MIDI Folder

 Samples Folder

The folder structure under the MuLab folder replicates how data is structured within the MuLab Project Browser, which I have elected to follow for consistency.

For example:



- Under **MIDI**, I have copied the MIDI clips exported from Cubase;
- Under **Samples**, I have copied the contents of the Cubase Mixdown Folder.

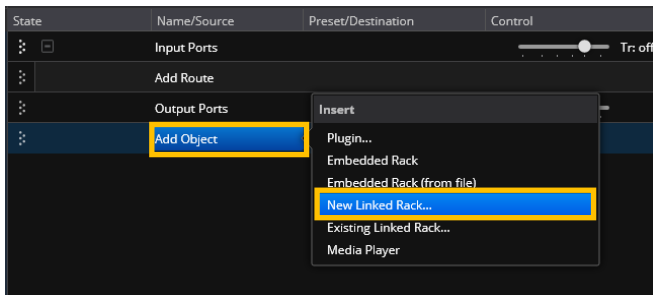
The reason I have done it this way is to make it easy to sync projects from my **DAWPC** to my **GIGPC**. As I prefer all data for a set to be under one file tree branch.

MuLab defaults to dropping data in the VST Plugin location, which I do not like myself.

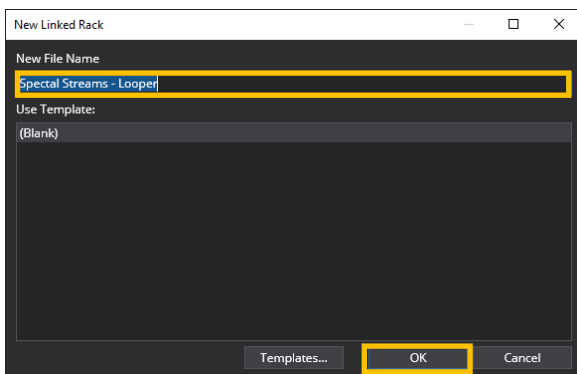
Creating the Cantabile Linked Rack for MuLab

Open Cantabile and the Setlist you are working with, and create a New Song and Save it if you wish if it is to be a test song you use later (adding it to your Setlist as well). Within this song we will create the MuLab Linked Rack. The song can be treated as a temporary song until much later on in this process, it is the Linked Rack that you need to take care to save as we go along. Or you can treat this initial song as one that you will use later.

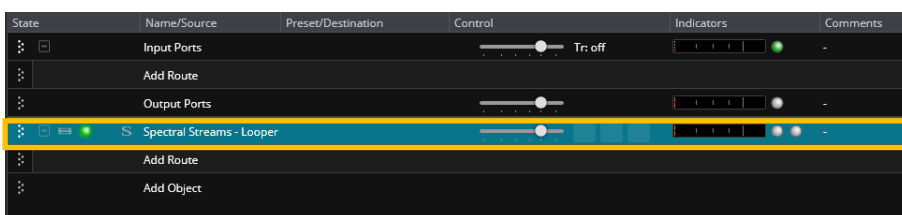
Click on **Add Object** and select **New Linked Rack...**



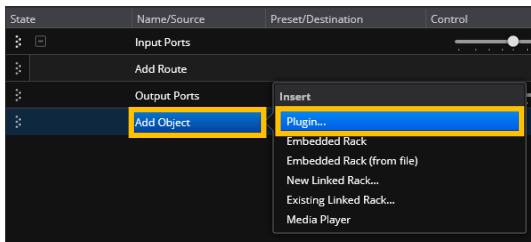
Give your rack a name and click **OK**. Spectral Streams is the name of my solo ambient project for which I am investigating looping as part of a new live show I am creating, hence the Rack name **Spectral Streams - Looper**.



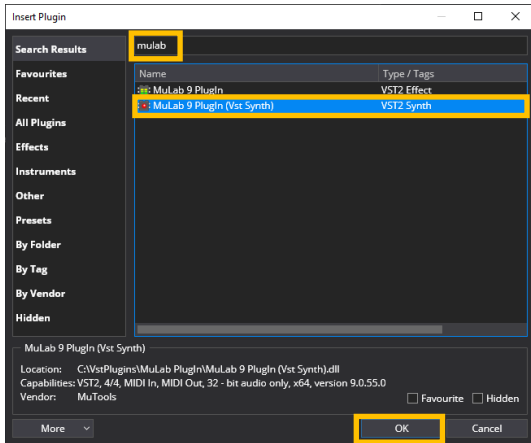
So now I have an empty linked rack, which I have coloured Cyan as shown below.



Double click on the rack to open it, and click **Add Object** and select **Plugin...**



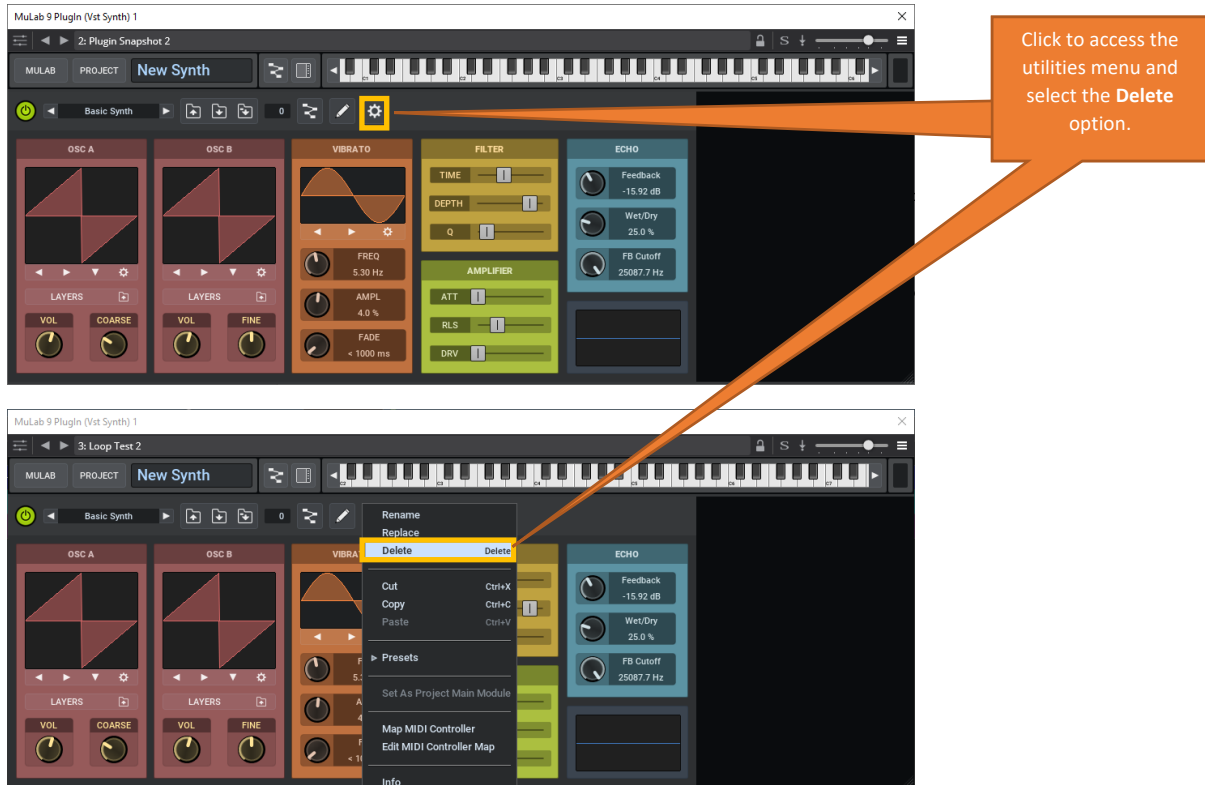
Filter the Plugin list by typing **mulab** and select **MuLab 9 Plugin (Vst Synth)** and click **OK**.



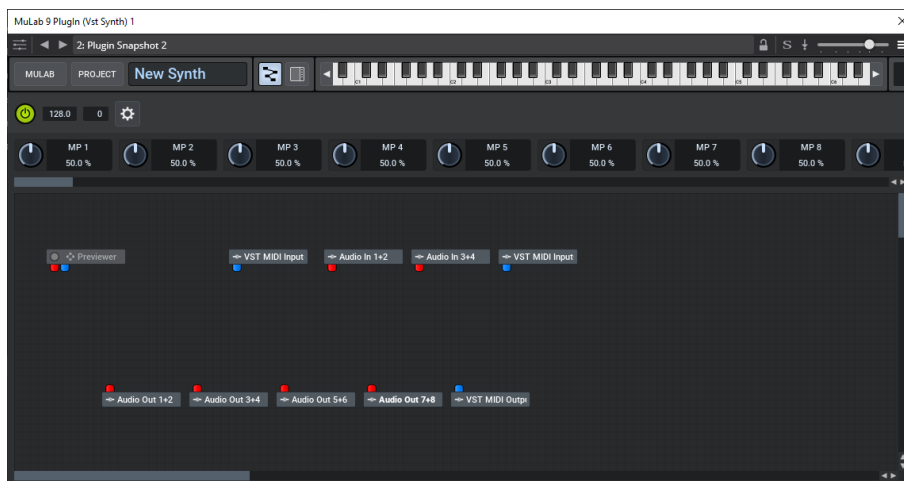
With that done, we now need to do a bit of work in MuLab itself. We'll return to the Linked Rack and how to configure it later on...

Creating the MuLab Project

Double click the MuLab object to open it. When you open MuLab, the default project is a “New Synth” that looks like this. MuLab has many different templates that can be used as starting points, including a Composer template, but I elected to start from scratch as the best way of learning.



Click over the gear icon (highlighted in yellow above) and select **Delete**. This gives you an empty project, which looks like this.



This is now ready for us to create our loop.

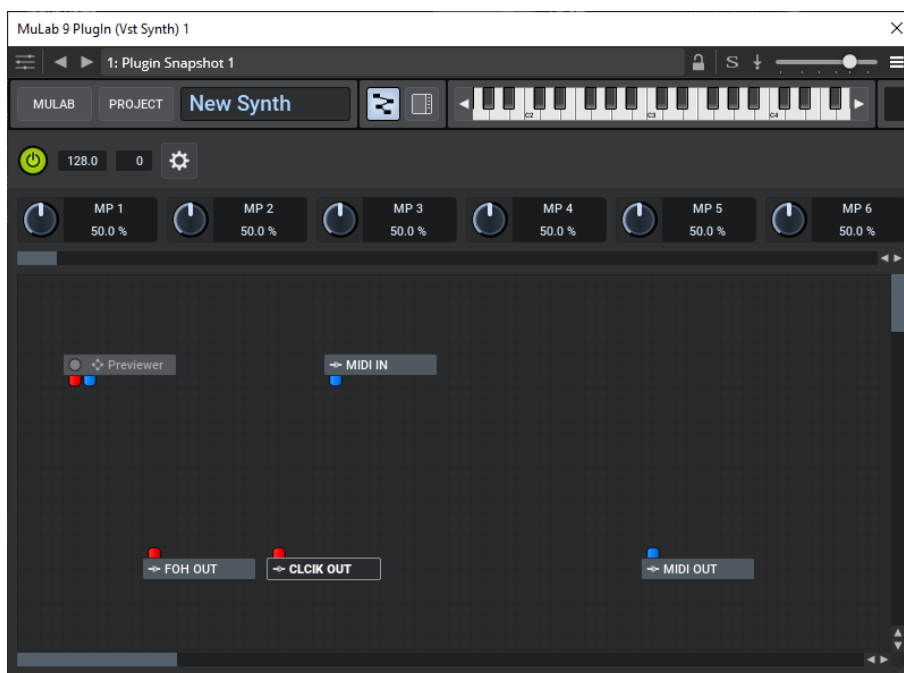
For my purposes, I deleted the following modules, by left clicking on them and pressing the DEL keyboard key (or right clicking and selecting **Delete**):

- The **Audio In 1+2** Module;
- The **Audio In 3+4** Module;
- The **VST MIDI Input** Module on the right;
- The **Audio Out 5+6** Module;
- The **Audio Out 7+8** Module.

I then renamed the following modules by right clicking over them and selecting **Rename** (you can of course leave them as is or choose your own names):

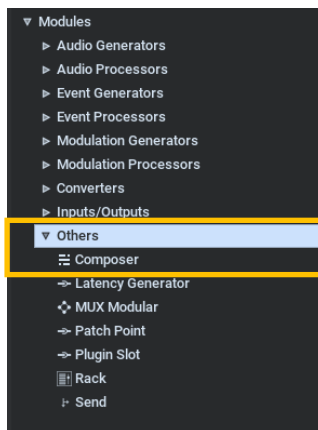
- **Audio Out 1+2** is renamed to **FOH OUT** (Front Of House, the sound that will go to the front of house mix along with other soft synths, etc.);
- **Audio Out 3+4** is renamed to **CLICK OUT** (my click track output that only the band will hear in their monitoring);
- **VST MIDI Input** is renamed to **MIDI IN**;
- **VST MIDI Output** is renamed to **MIDI OUT**.

This leaves us with the following:

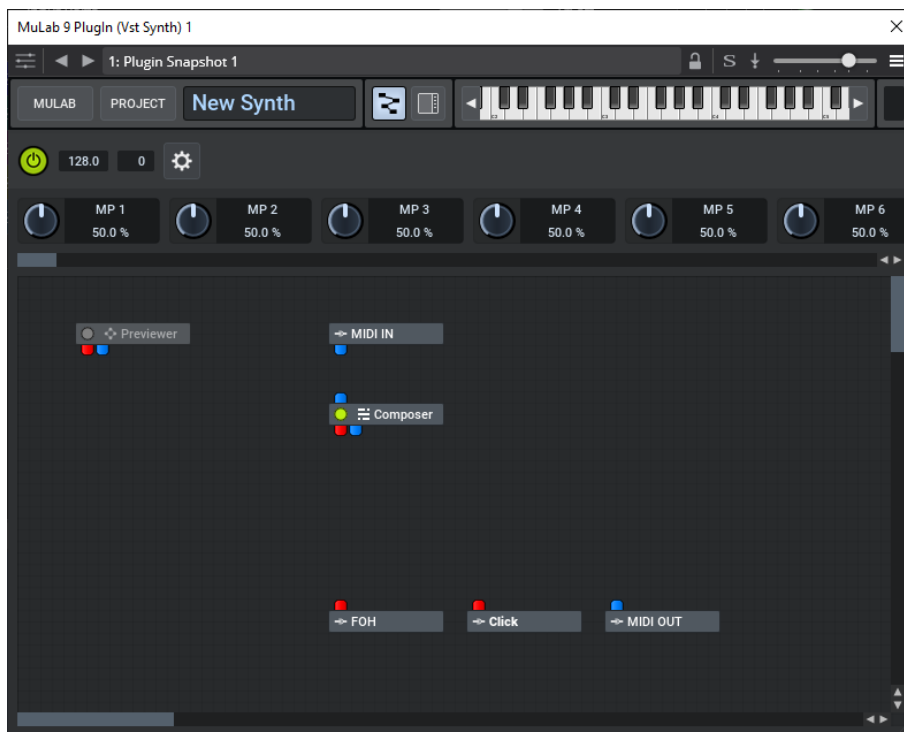


Next, right click over the window background and select **Add Module** from the context menu.

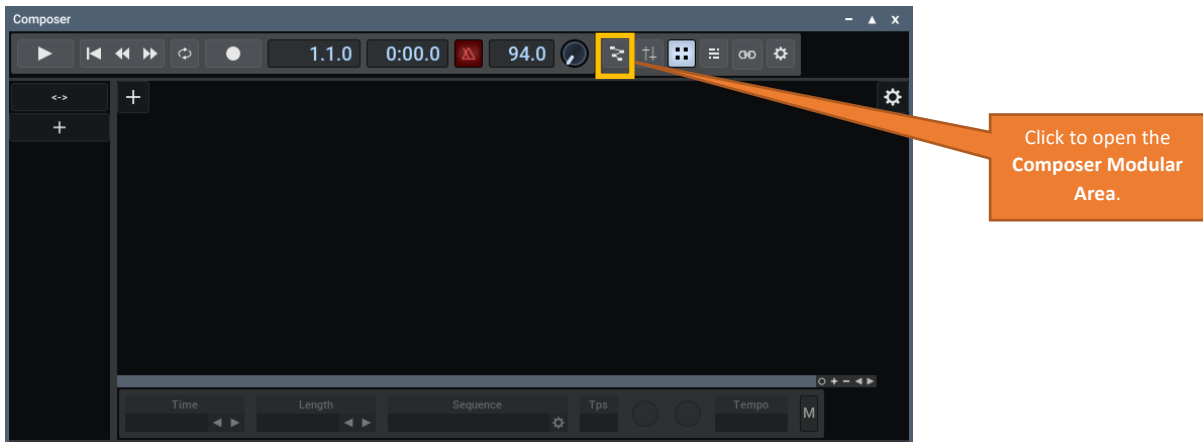
Expand the **Others** menu option and select **Composer**.



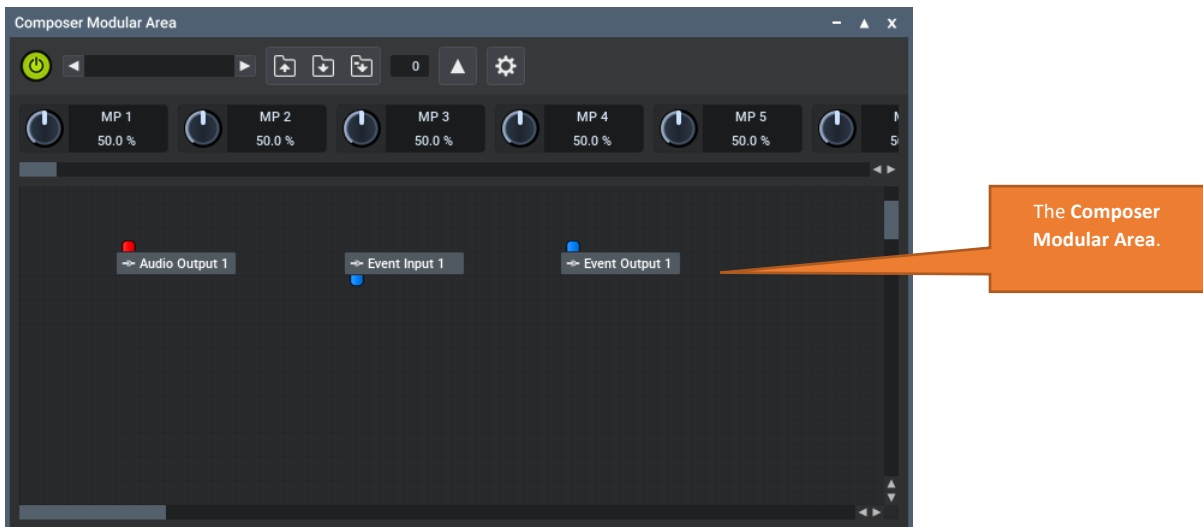
Arrange the modules as follows, or according to your personal taste.



Double click on the Composer module that you just added, and you will get a blank composer as follows.



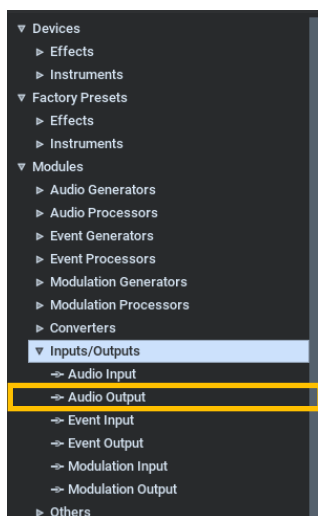
Click on the **Composer Modular Area** button in the tool bar (highlighted in amber in the above diagram), and you will get the following view.



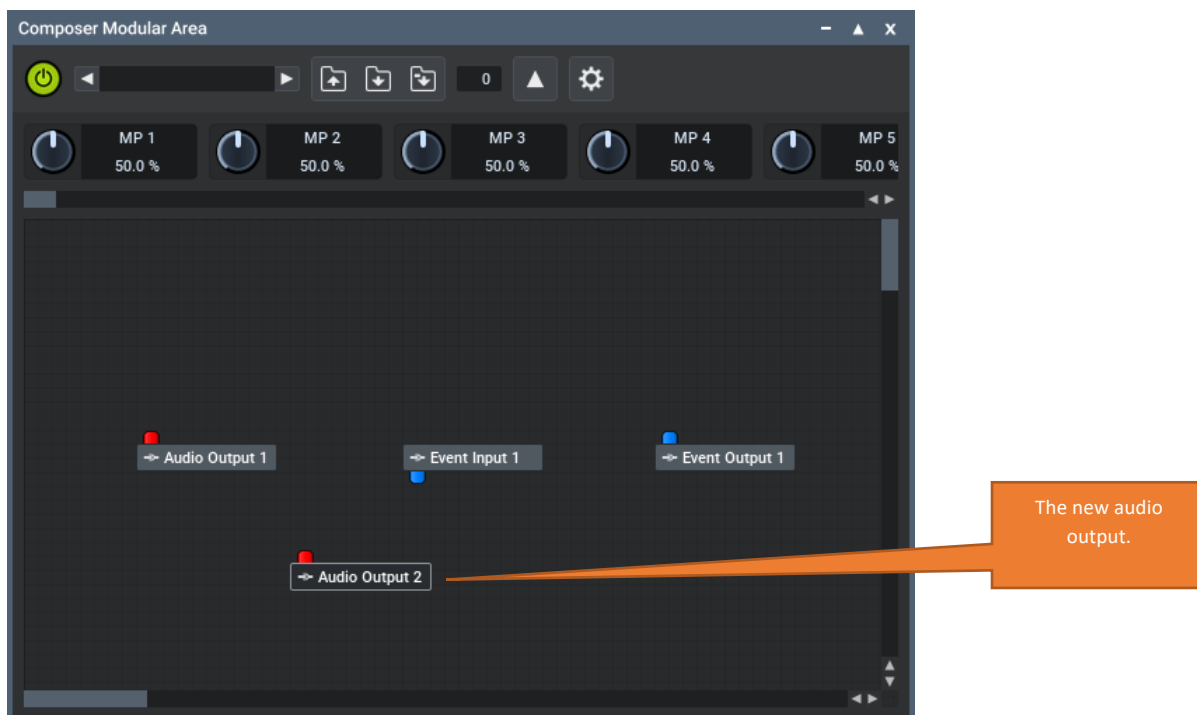
I want an extra audio output channel for the click track that I will be adding.

Right click over the background and select **Add Module**.

Under **Modules**, expand **Inputs/Outputs** and select **Audio Output**.



And you now have an extra audio output.

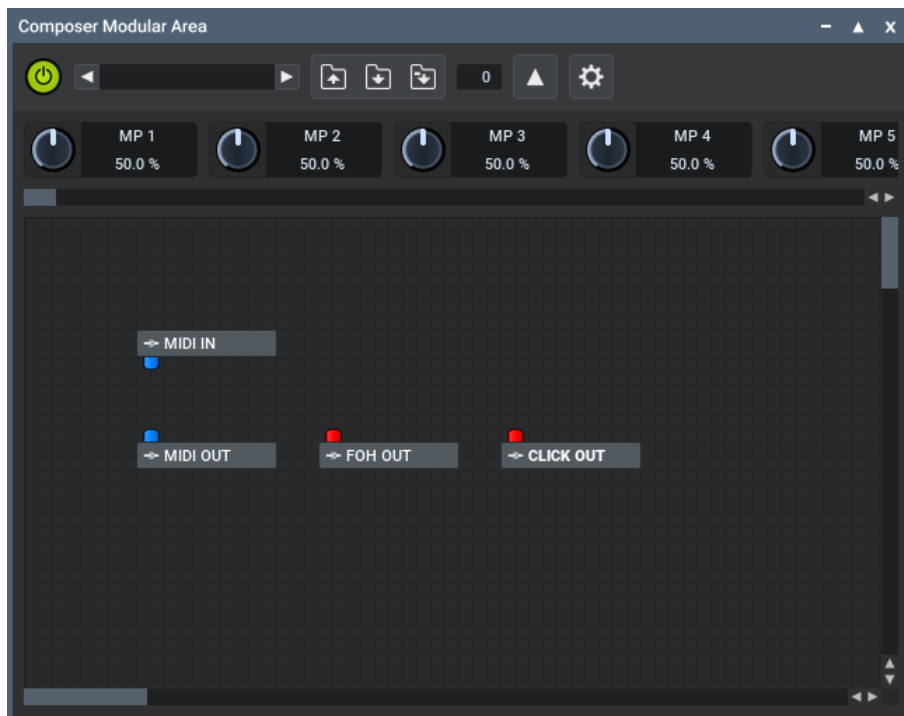


Rename the modules as follows by right clicking over each module and select **Rename**:

- **Event Input 1** to **MIDI IN**;
- **Event Output 1** to **MIDI OUT**;
- **Audio Output 1** to **FOH OUT**;
- **Audio Output 2** to **Click OUT**.

Or you can of course rename these according to your taste.

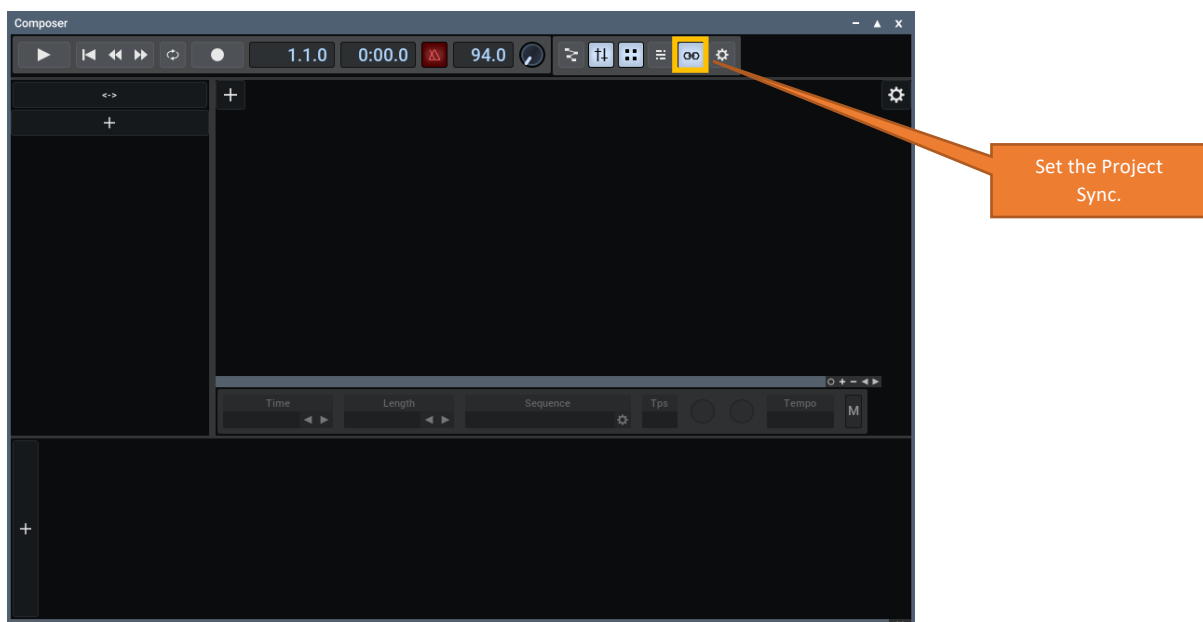
And you should have something similar to the modular area shown below, where I have reorganised things a little, but this is not essential.



You can now close the **Composer Modular Area**.

Next set the MuLab **Host Sync** (yellow highlighted button shown below) so it is selected.

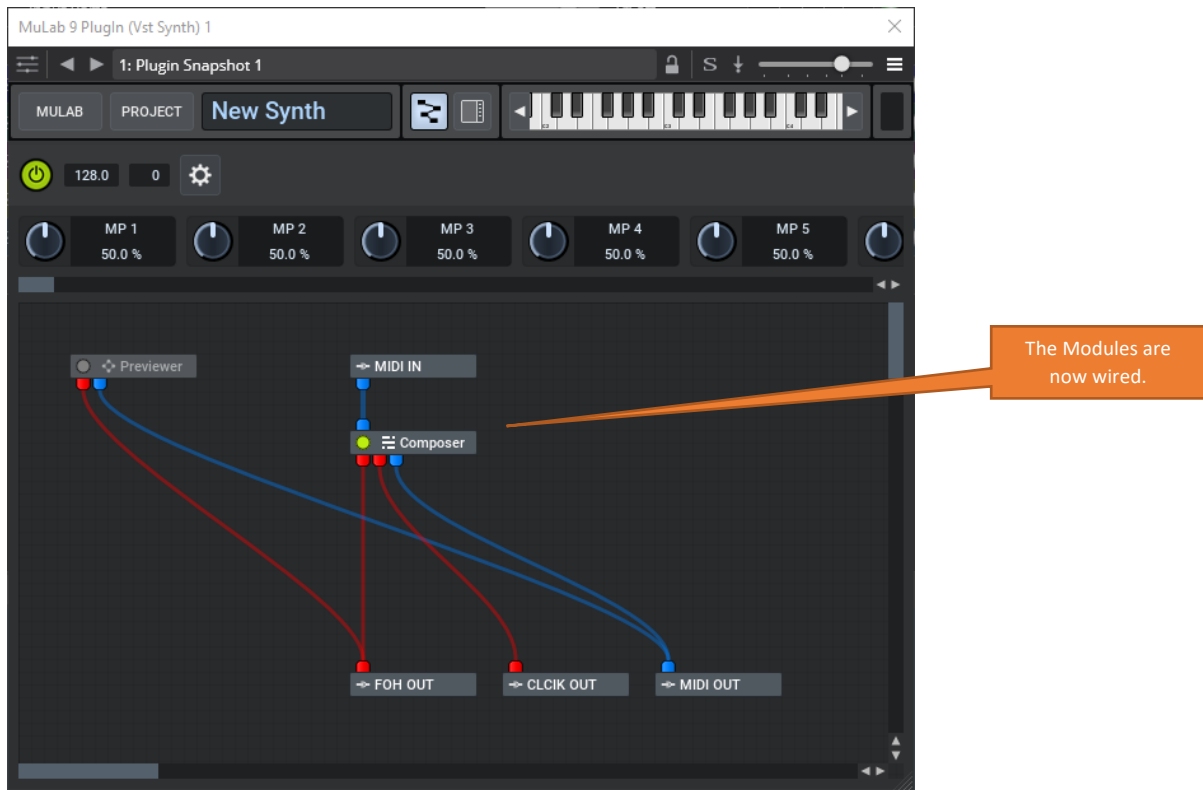
This means that MuLab will take the tempo and Bar/Beat location from Cantabile.



You can now close the **Composer Window**, to take you back to the main MuLab Window where you can now wire the modules.

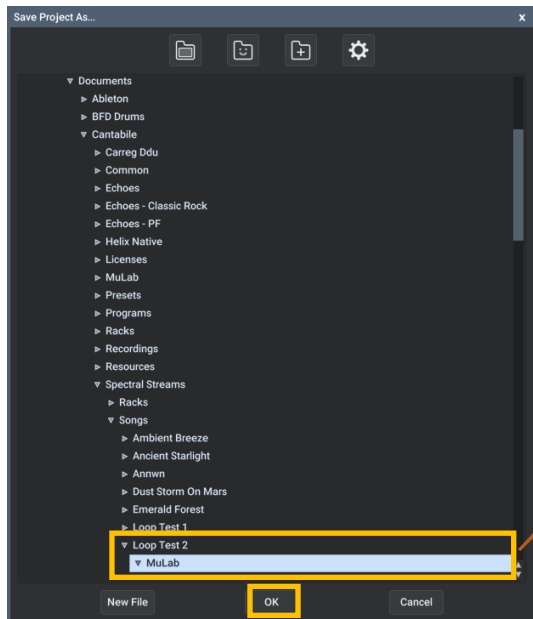
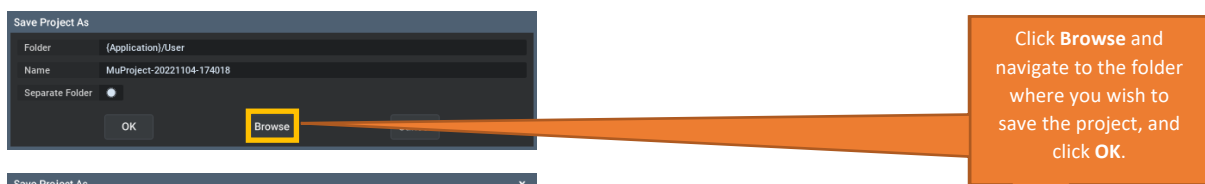
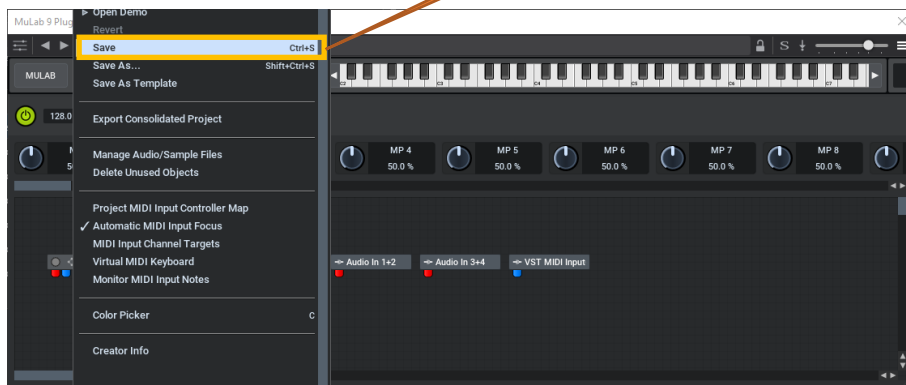
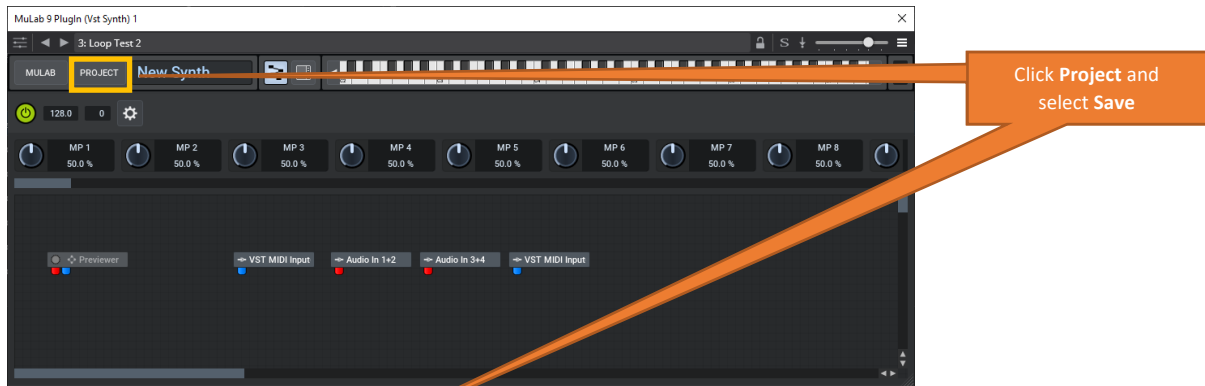
To wire the modules, left click on a source port (red for audio and blue for MIDI) and drag and release on a destination port.

You should now end up with the following.

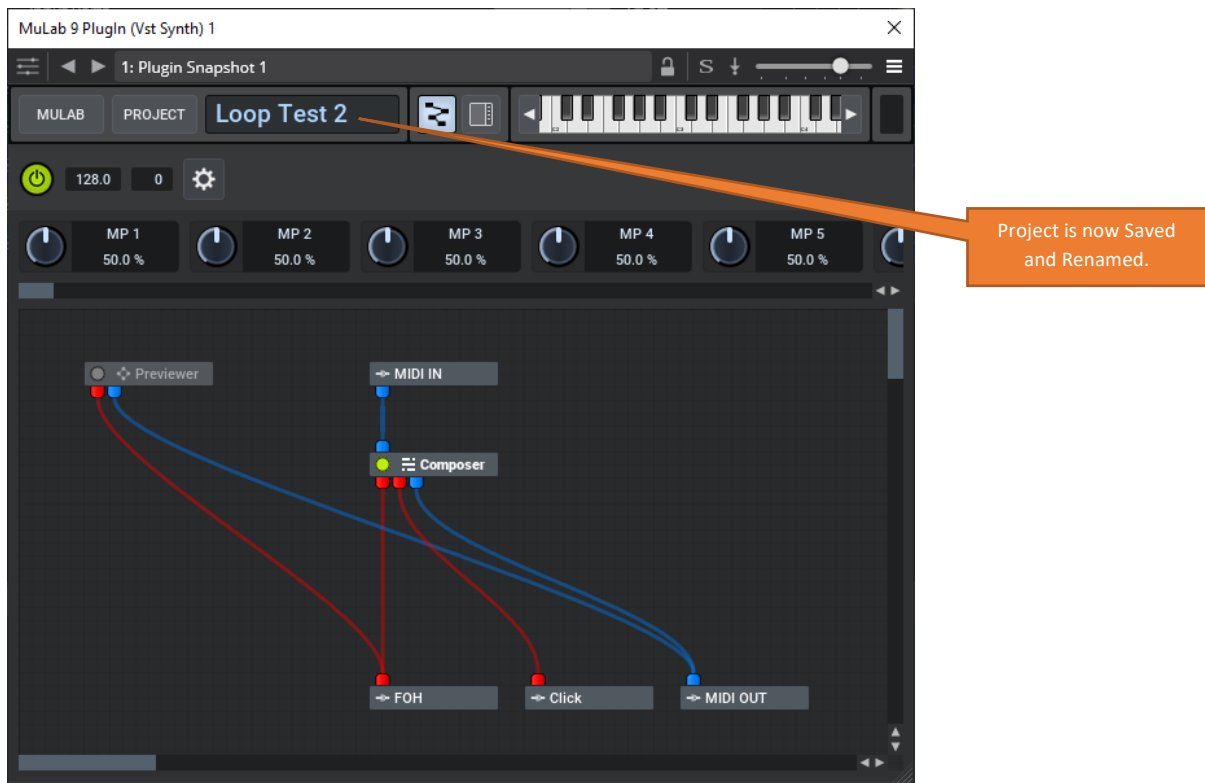


I did delete the **Previewer** module in my early tests, but it keeps coming back if you play samples in the MuLab Project Browser when you get that far, so I just left it alone in the end.

Now is a good time to save the project. Click on the **Project** button in the tool bar and select **Save** and follow the sequence below. As this is a new project **Save** will act the same as **Save As...**

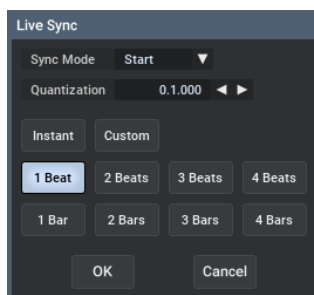


You can see below that the project is now saved with my chosen name. This example I am covering is my second loop test hence the rather boring name!



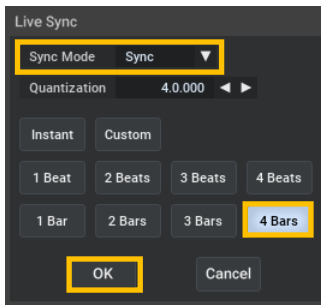
Next, we need to set the sync mode which determines how quickly MuLab changes to a scene when you wish to make the change. For my project, I want all loops when active to complete playing before the next loop starts.

Right click over the Composer Module and select **Live Sync Mode**, which brings up the following dialog.

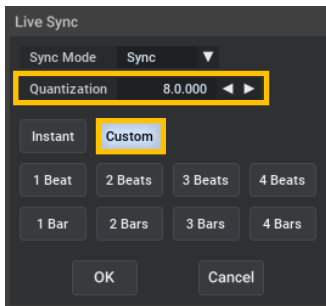


I want it so that changes only occur at the end of a loop, and I am making my example loops multiples of four bars – this may vary depending on the song/loop and you can of course vary it according to your preferences.

So, change the **Sync Mode** to **Sync**, and **Quantization** to **4 Bars** and click **OK**.

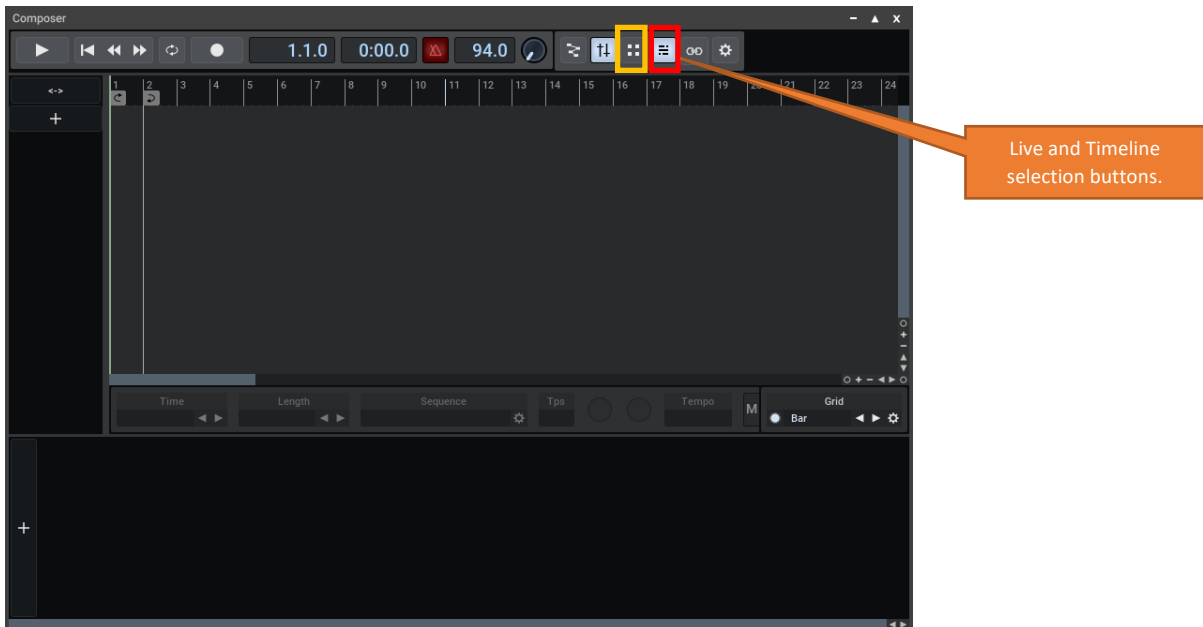


Note how selecting a button changes the Quantization value. You also can have different values to standard ones provided. For example, 8 Bar Quantization looks like this:



This sets the “global sync”, which you can override within the loops if you need to, for example if some loops are different lengths.

We are now ready to build the looper module in MuLab. Double click on the **Composer** module again to open it, as we are going to do a bit of work in that, and it opens like this in the timeline view (so it looks like a conventional sequencer).

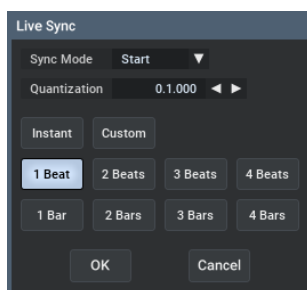


However, we want to work with the live matrix view, not the timeline view. So, we need to turn on the **Live Matrix** view (yellow highlight above) and turn off the **Timeline** view (red highlight above), which gives us the following view.

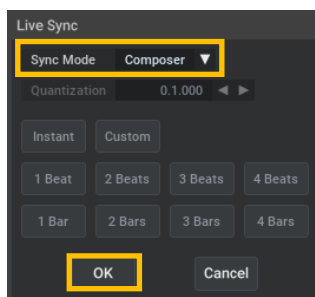


Note that one view must always be active, hence the order is to turn on the **Live Matrix** view before turning off the **Timeline** view.

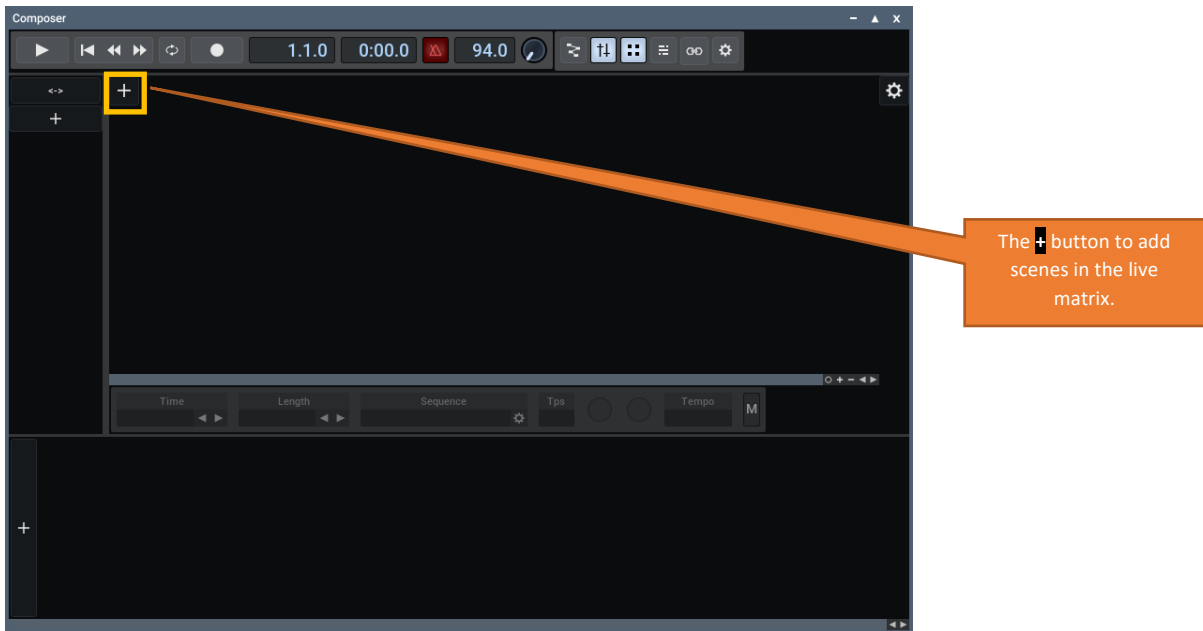
Now, click on the gear icon (yellow highlight above) and select **Live Sync Mode**, which brings up the following dialog.



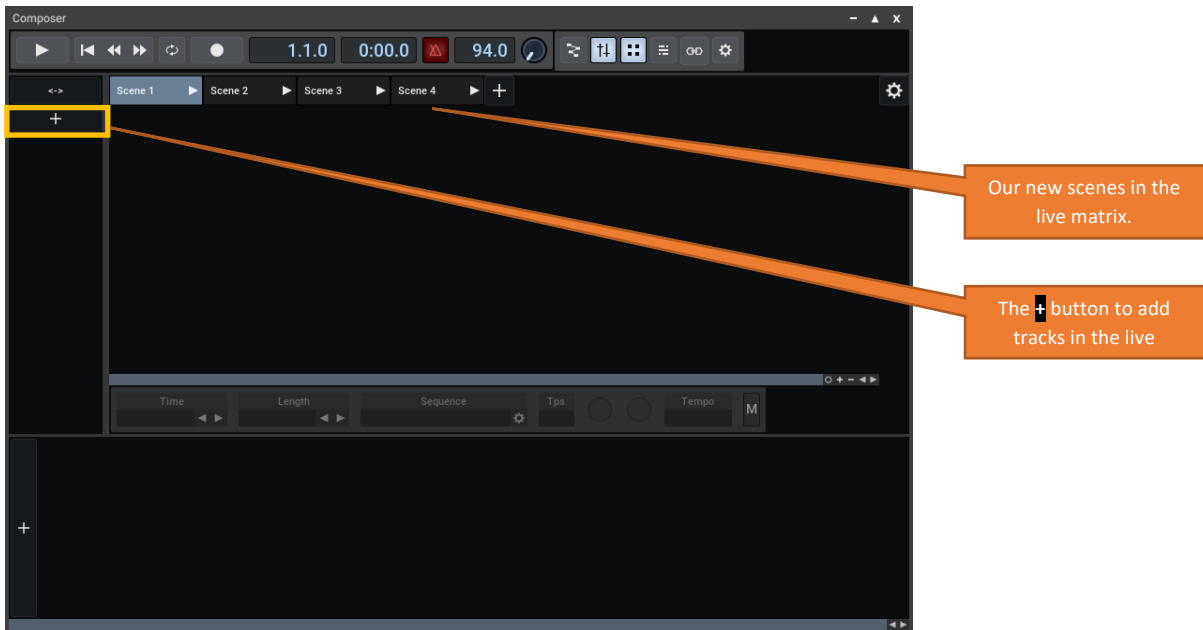
For the timeline view, we want it to pick up the Composer Module Sync settings, so under **Sync Mode**, select **Composer** and click **OK**. This means that the live matrix will use the Composer Sync Settings that you set up earlier.



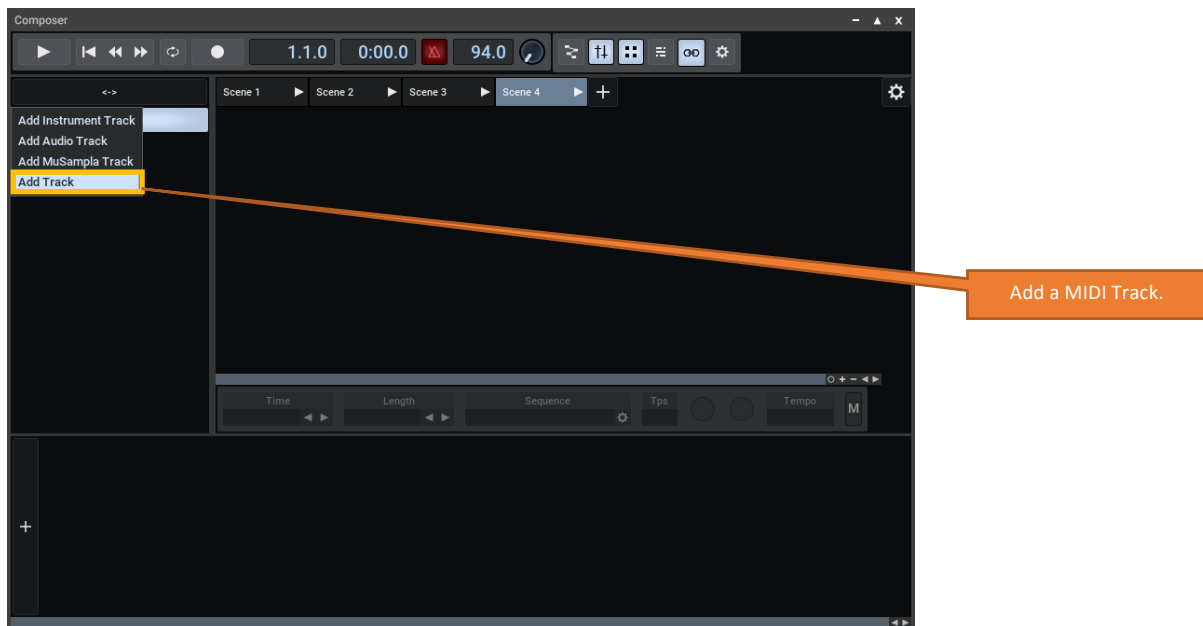
Following this, we can now add the Scenes that we wish to loop around. I am creating this demo song with four scenes, so click the **+** icon highlighted below four times to create the four scenes.



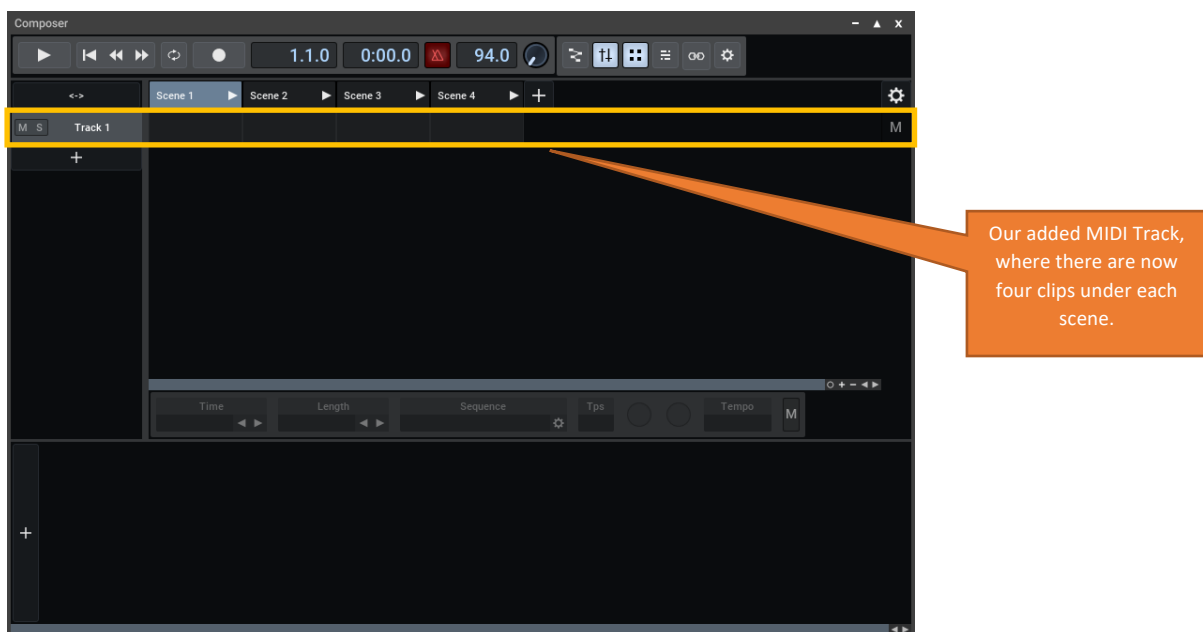
And you should now have the composer like this, ready to create the tracks, using the highlighted button below.



First, we will create the scenes for the MIDI track, so press the highlighted **+** icon shown above and select **Add Track**, which adds a MIDI track.



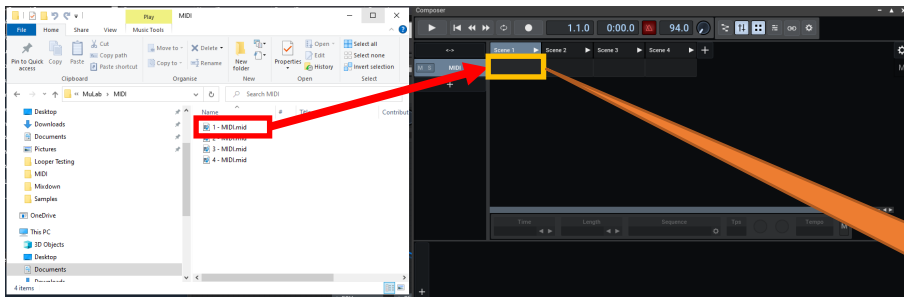
Once this is done, we have the following empty track created.



Using the right click context menu over **Track 1**, make the following changes:

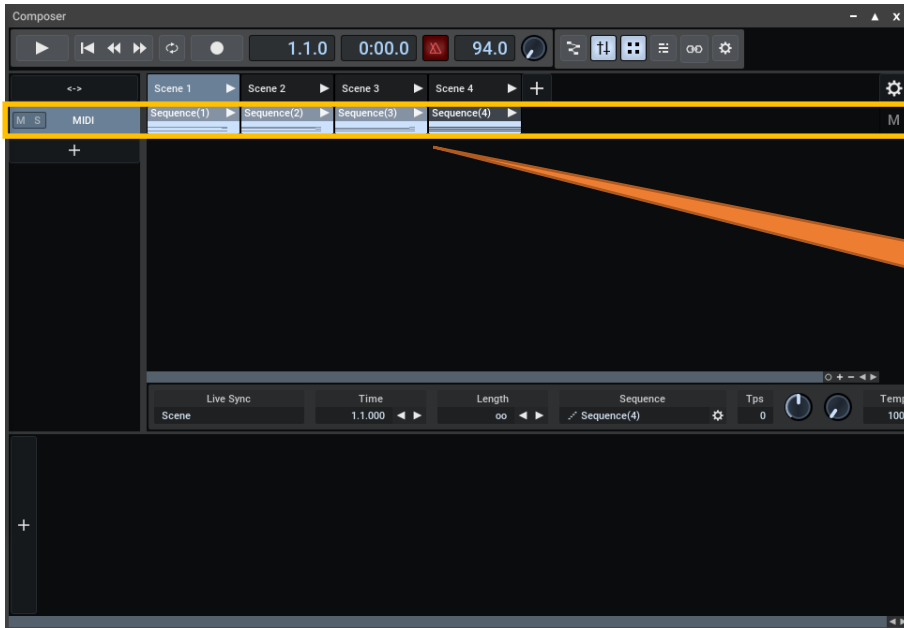
- Select **Rename** and rename the track to **MIDI**;
- Select **Choose Target Module** and set to **Composer/MIDI OUT**;
- Select **Choose MIDI Channel** and set to **Per Clip**, which will use the MIDI channel settings of each MIDI clip (which you will set later).

You are now ready to setup the MIDI clips created earlier. To do this you can open Explorer, navigate to your MIDI clips and drag and drop the four clips into MIDI track on the composer view, with each MIDI file being dragged to the respective Scene.



Drag and drop each MIDI file to each clip, starting with file 1- MIDI to Scene 1, Track 1.

You should end up with the following.

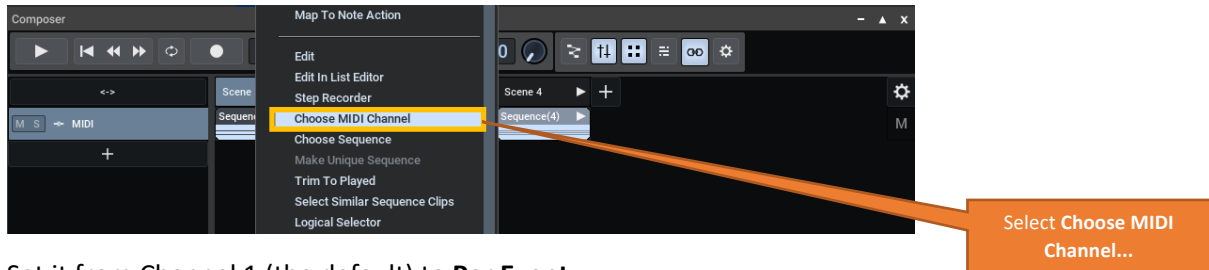


All of our Clips on Track 1 now have the MIDI files inserted.

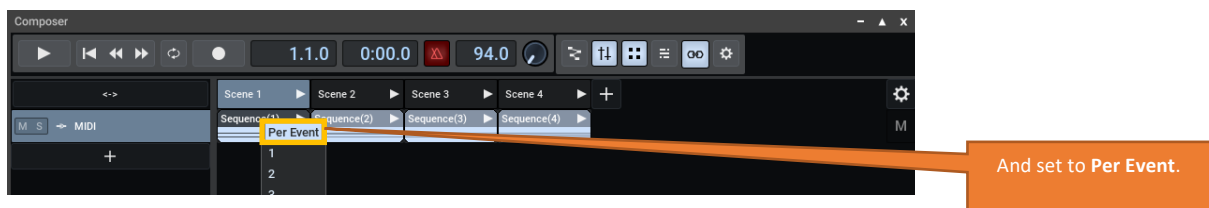
The MIDI channel for a Clip is set to Channel 1 by default, so all MIDI data in the clip irrespective of the recorded channel number is going to be mapped to channel 1, which we do not want in my example.

If you have MIDI data in different channels that you want to come out on those channels as recorded, then you need to make the following change.

Right click over a MIDI clip and select **Choose MIDI Channel**.



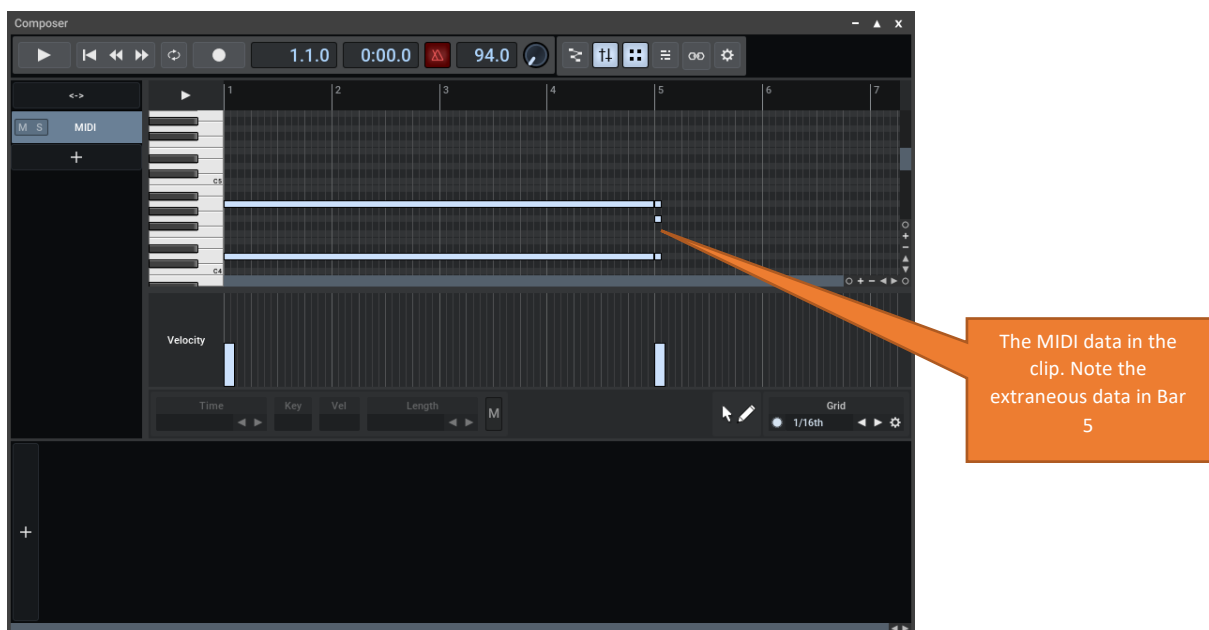
Set it from Channel 1 (the default) to **Per Event**.




Repeat for all Clips. You can also do this from within the editor context menu in the steps below.

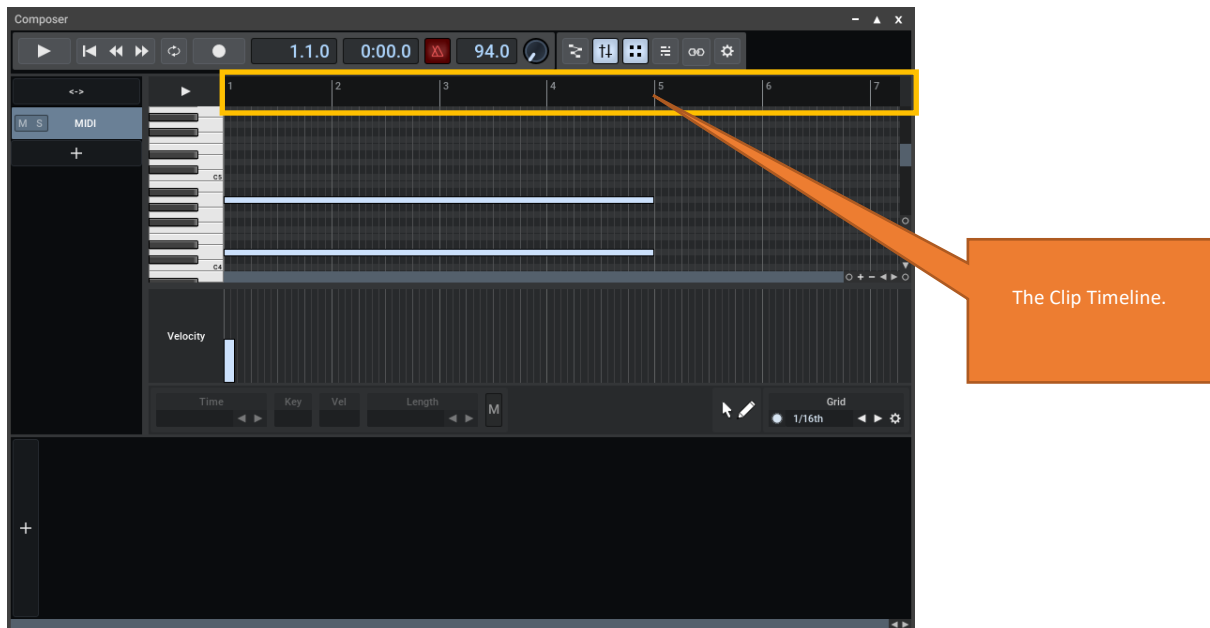
You now need to set up the looping in each clip.

Double click on **Sequence (1)**³, and you will get the following view (of course your MIDI data can be quite different to mine; I have some simple block chords in my clips to play an arpeggiated sound in Dune 3).



³ Or you can also single click the clip and press the  Button.

The fact that I have MIDI data on exact quantized boundaries means that some data from the next bar has been picked up in the Cubase export. Whilst it is probably not essential, it is easy to delete this by selecting the data you do not want and press the DEL key. In the picture below you can see that the short notes seen at the start of Bar 5 in the previous picture have been removed.



If you right click over the timeline area (highlighted in yellow above), at the Bar 1 location, select **Set Clip Start** and then right click again and select **Set Loop Start**. At the bar 5 location, right click and select **Set Clip End**.

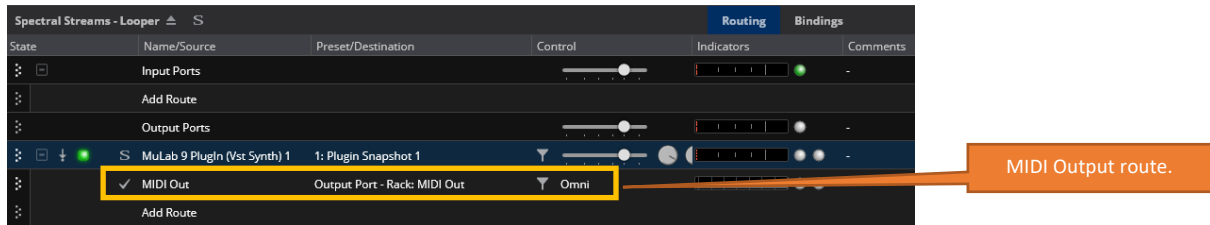
You should now see the loop markers on the timeline.



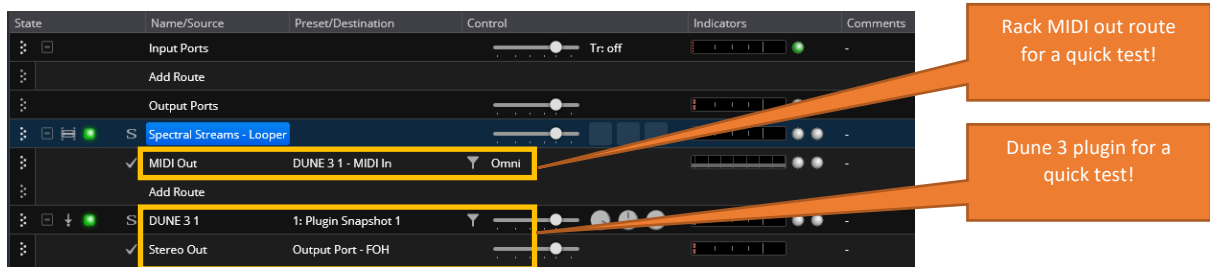
To get back to the main composer Window, click the **<->** button (highlighted in yellow above) and repeat the steps above for all the MIDI clips.

At this point, we should have done enough to do some manual playback if you wish to see some interim results! Skip this step and go straight to **Creating the MuLab Audio Tracks** if you wish to save it all until the end!

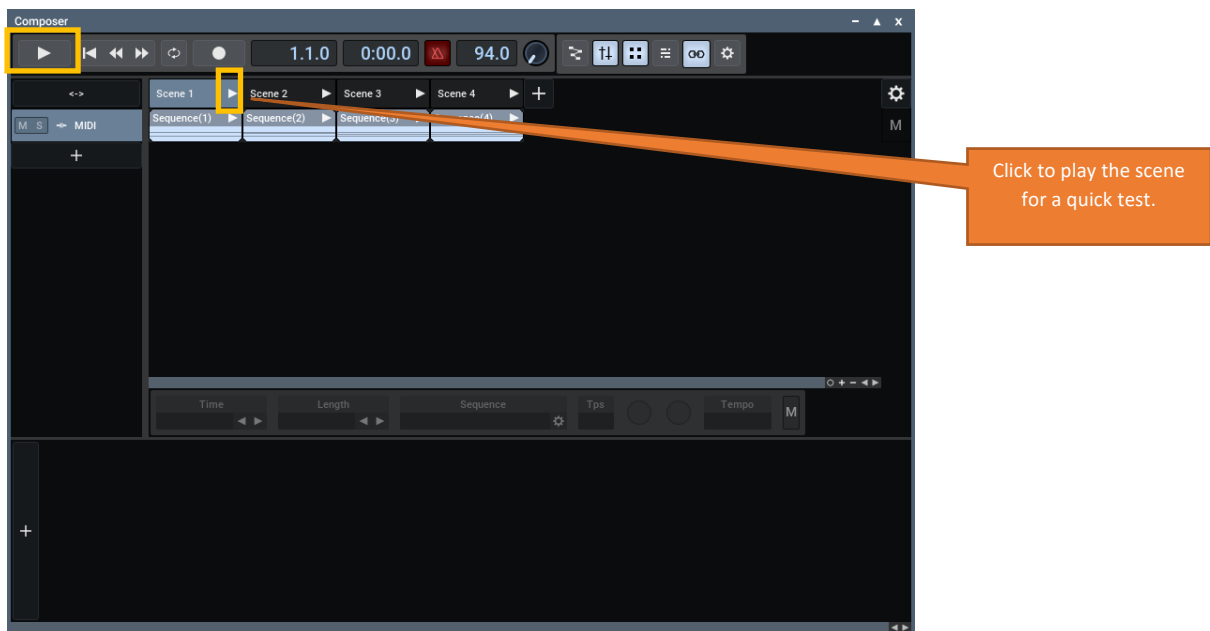
Back in the Cantabile Rack, create a **MIDI Out Route** to the rack's **Output Port – Rack: MIDI Out**.



Close the Rack and add your plugin of choice to the Cantabile song you are creating the rack in, setting up a MIDI route from the Rack to the plugin, and of course set the plug in so that it has an audio route to your audio output.



If you now open the rack again and go into MuLab and the Composer module, and press either the Scene 1 play button or the master play button, you should have your VST (Dune 3 in my case) playing!

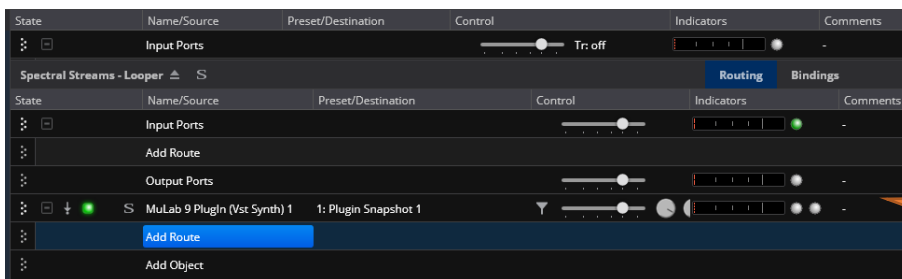




If you now click on a different scene, it should not start playing until the next four bar boundary (or whatever quantize you set). If this does not happen as expected, then you need to check your settings.

You can also trigger the clips individually if you wish, but for this tutorial, I am working at scene level.

Once you have done this test, delete the Plugin within the song, and the MIDI Out route within the rack, so we are back to the following, as later we will have a lot of rack configuration to do.

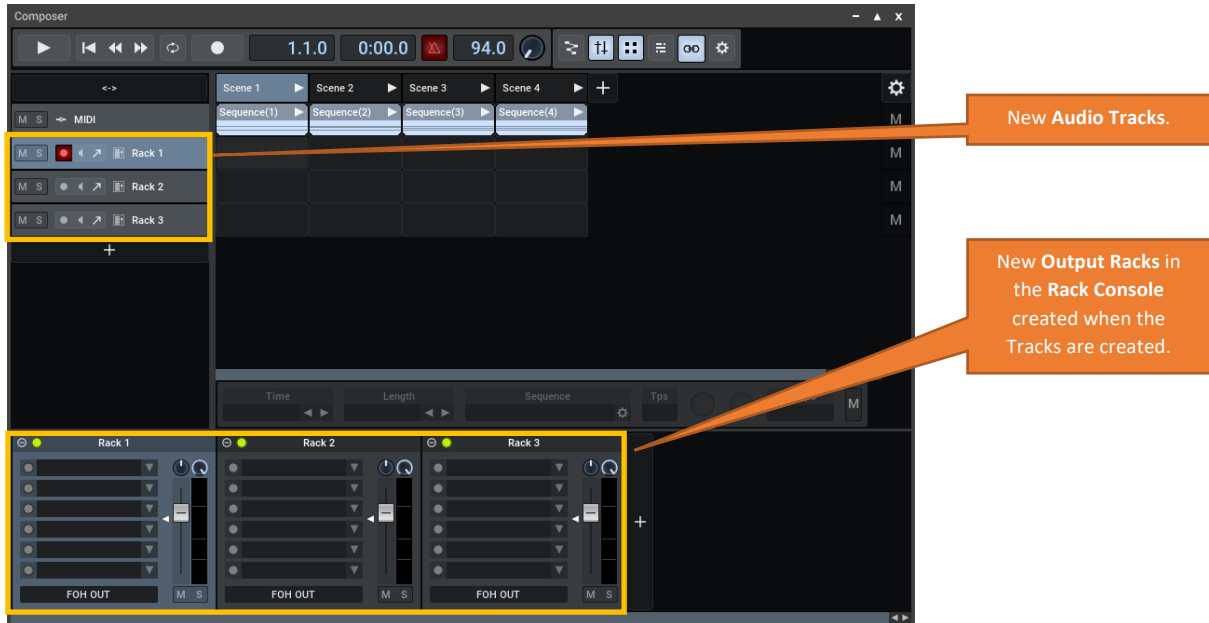


Temporary routes deleted, ready to do it properly!

Next, we will set up the audio tracks and audio clips within the scenes.

Creating the MuLab Audio Tracks

Click the **+** Button below the MIDI track three times to add three audio tracks using the **Add Audio Track** option this time for each track, and you should get the following view, noting that I dragged the vertical divider to the right to see the names of the tracks.

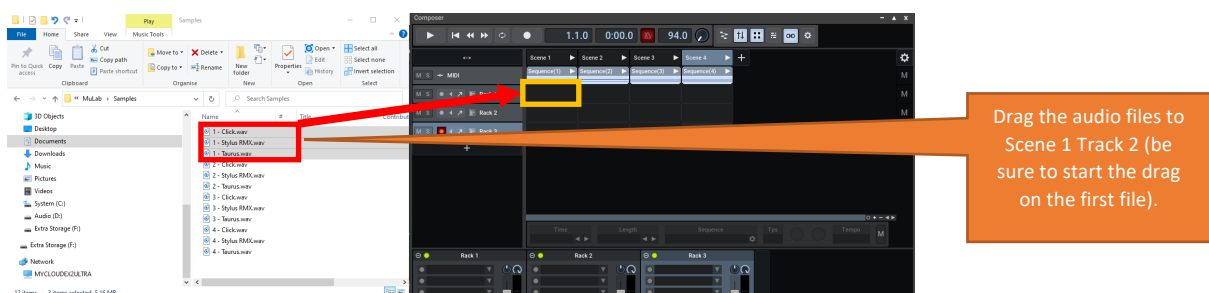


Each track also gets its own processing rack in the **Rack Console** at the bottom of the window, and each track is named after its corresponding Rack.

We are now ready to add the audio clips to the tracks.

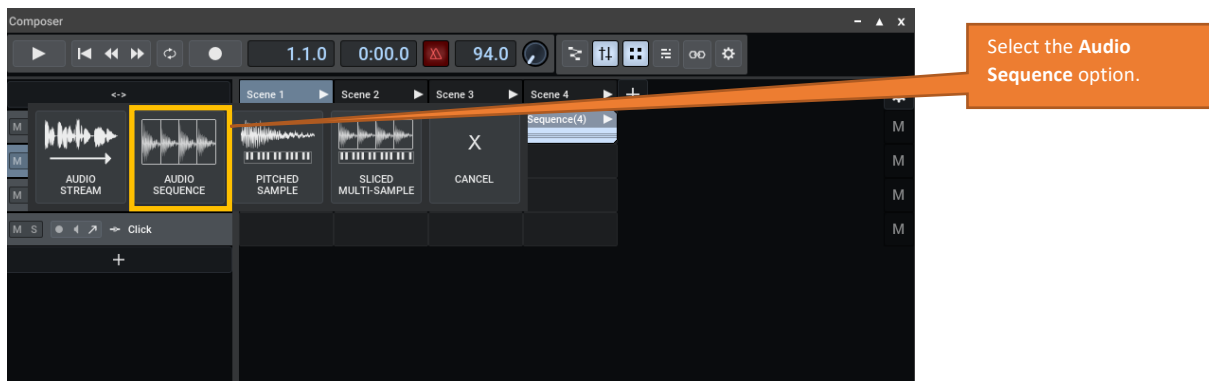
Navigate to the folder where you are storing the audio clips.

Select the 3 clips for scene 1 and, clicking on the first clip in the selected group⁴, drag them to Track 1, Scene 1.



⁴ The clip you initiate the drag over will be the one dropped first in the scene, so it is important that you start the drag on the first one.

Before the waves are inserted into the clips, you will be asked to select what you want the clips to be as shown below. Select **Audio Sequence** (highlighted in yellow below).



You could of course have dragged the clips individually into the Scene, but copying three at the same time into the scene is a neat shortcut.

Repeat the drag and drop for the remaining scenes.

Right click over each **Rack**, and rename as follows:

- **Rack 1** is renamed to **Click Track**;
- **Rack 2** is renamed to **Stylus RMX**;
- **Rack 3** is renamed to **Taurus**.

Note that the tracks have also been renamed.

And set the **Click Track** Rack output to **CLICK OUT** by right clicking over the output port name at the bottom of the rack and select the required port.

And because I like the order

- Bass;
- Drums
- Click

I have dragged the tracks and modules about as shown below. I could have set this order up first of all, but then I would have had to drag the individual clips.



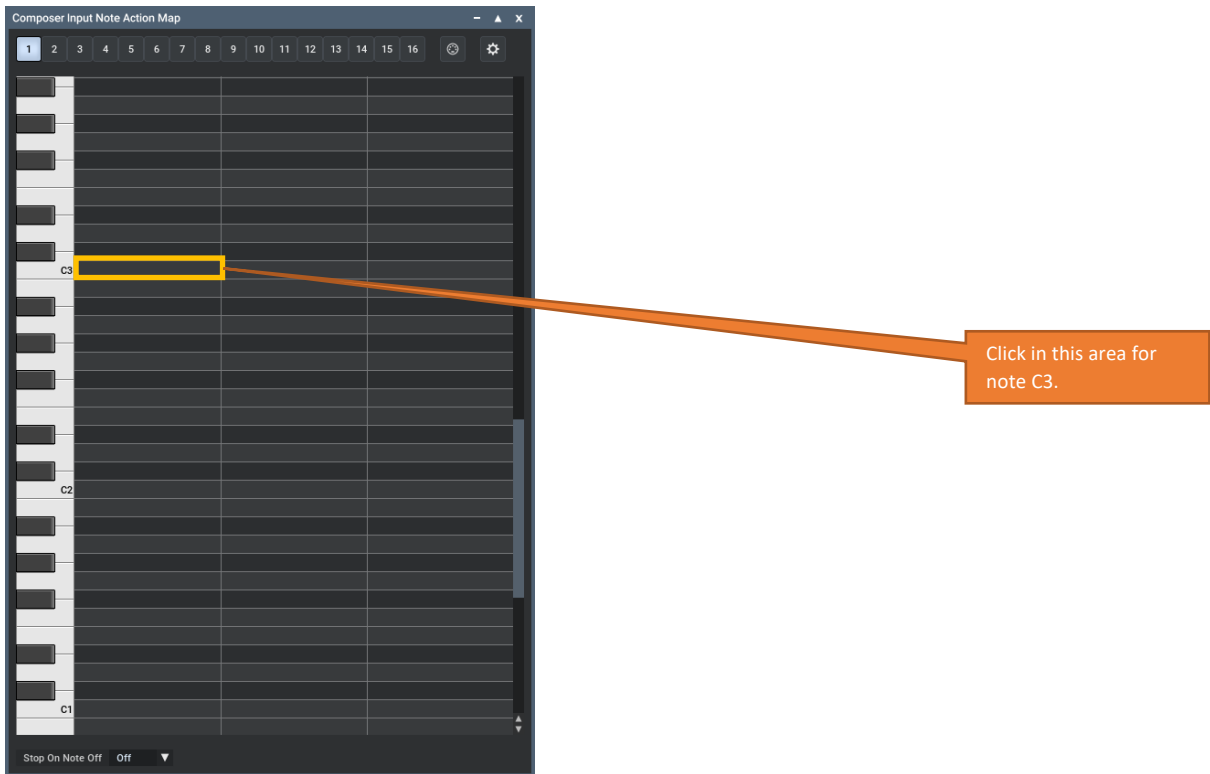
This is a good time to save the project again!

The next step is to be able to trigger playback of the Scenes, for which you will need a **Note Action Map**. This is the only way I have found so far to trigger the Scenes in MuLab.

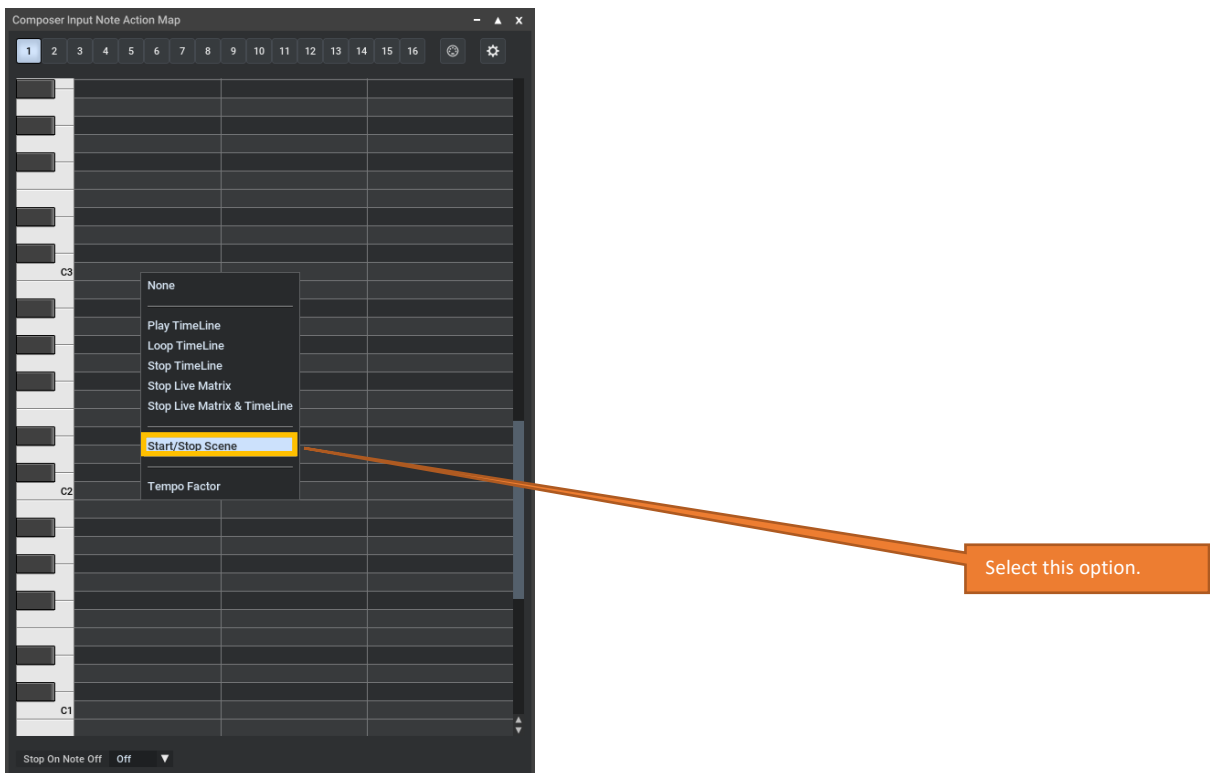
Click on the Gear Icon in the Composer window (yellow highlight in the diagram below) and select **Note Action Map**.



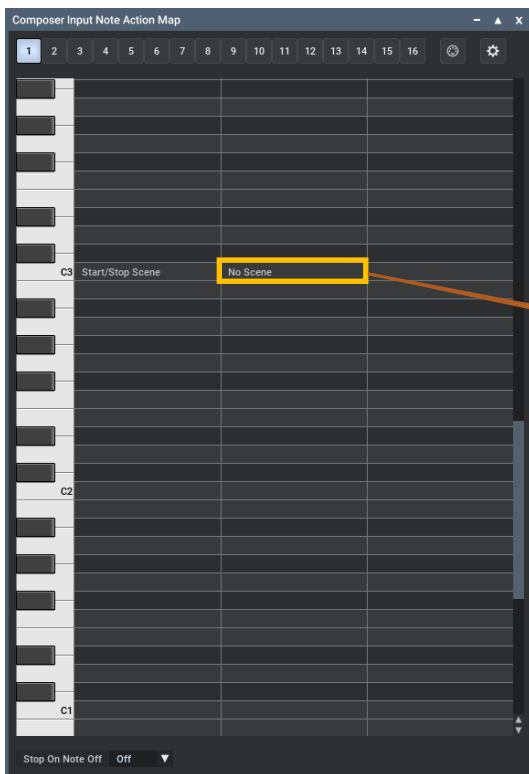
Here you can set up the note triggers for each Scene (and for doing other things if you wish). I am going to build my note action map starting at note C3.



For the note you wish to trigger a Scene from, left click in the first area (yellow highlight above) for Note C3, which will be my Scene 1 trigger, and select **Start/Stop Scene**.

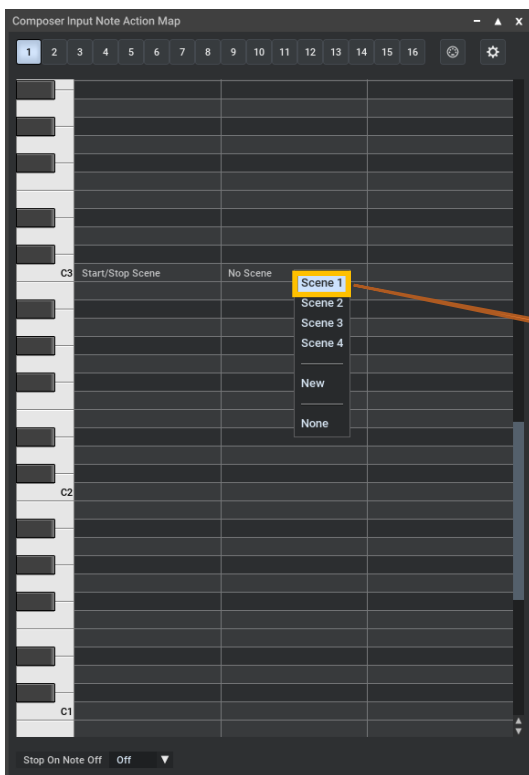


You will now get the following



Now, click in this area.

Left Click over **No Scene**, and select **Scene 1**.



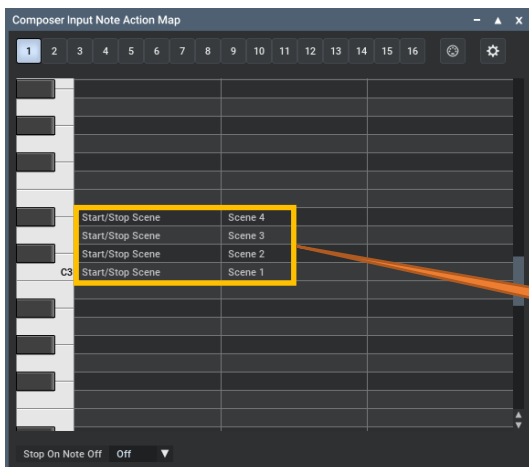
And select **Scene 1**.

Which gives you the following.



Key C3 now has an action to start/stop Scene 1.

Repeat for remaining scenes. You can of course use whatever note triggers that you wish to use.



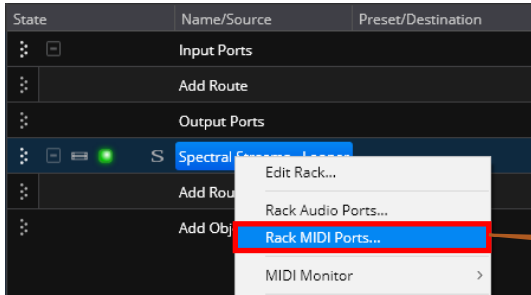
We now have note actions for all four scenes from C3 to D#3.

Save your project again, as (finally!) that is all of the main work we need to do in MuLab, and we now need to setup Cantabile.

Setting up the Cantabile Linked Rack for MuLab

After all that hard work, we are now ready to setup the Looper Rack to encapsulate the looper functionality. This is of course how I wish it to work in my setup, you may need to tailor this to suit your own workflow, but I would suggest you follow the example and consider how to tailor it later.

Right click over the Rack and select **Rack MIDI Ports...**



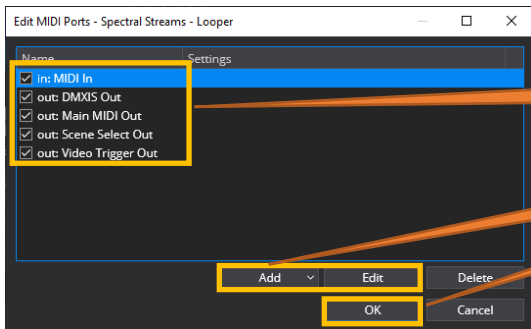
Select this to set up the Rack's MIDI ports.

Edit **Midi Out** and rename it **Main MIDI Out**.

And also **Add** the following MIDI Output Ports:

- DMXIS Out;
- Scene Select Out;
- Video Trigger Out.

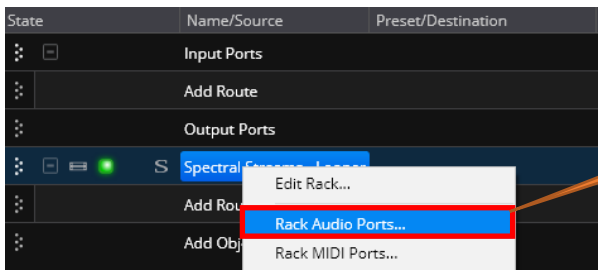
And you should have the following.



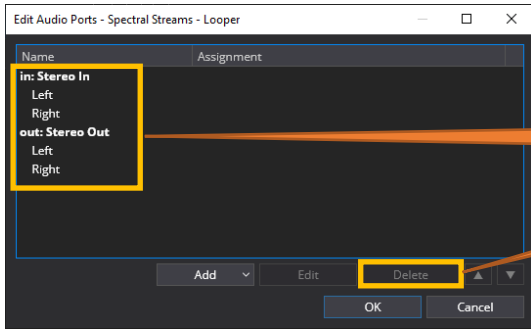
Use Edit/Add to setup these ports and then click OK.

Click **OK** when done.

Right click over the Rack and select **Rack Audio Ports...**



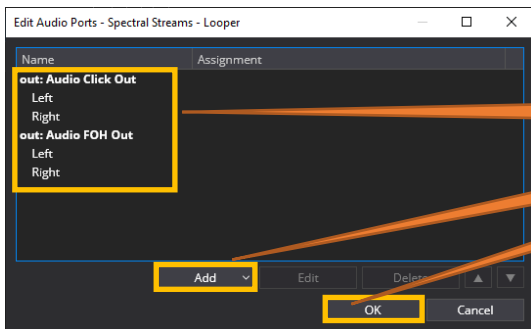
Now select this option to setup the audio ports for the Rack.



Use the **Delete** option to delete these ports.

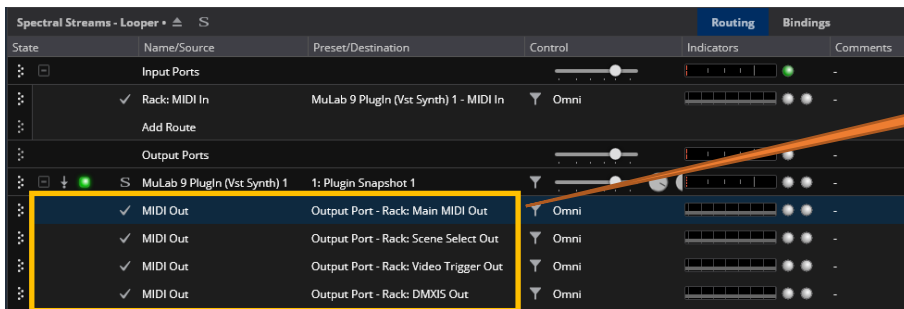
Delete the existing ports and **add** the following **stereo output ports**, and click **OK**.

- Audio Click Out;
- Audio FOH Out.



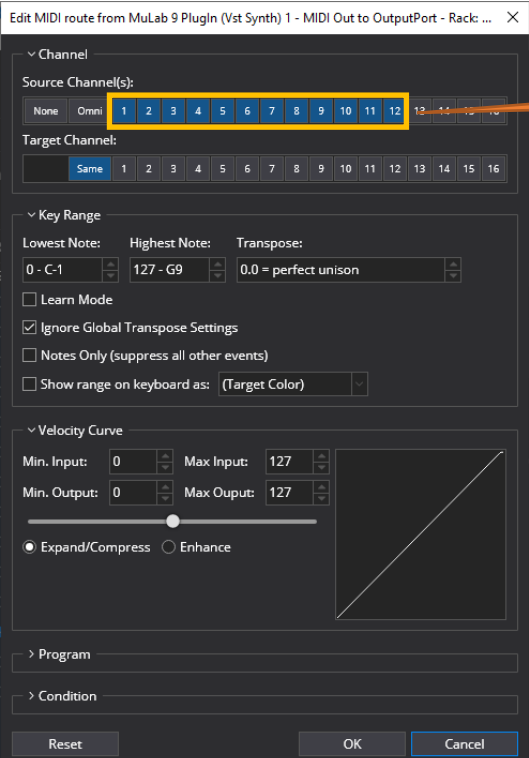
Use the **Add** option to add these ports and click **OK**.

Now open the Looper Rack again and under the MuLab plugin, add the following routes:



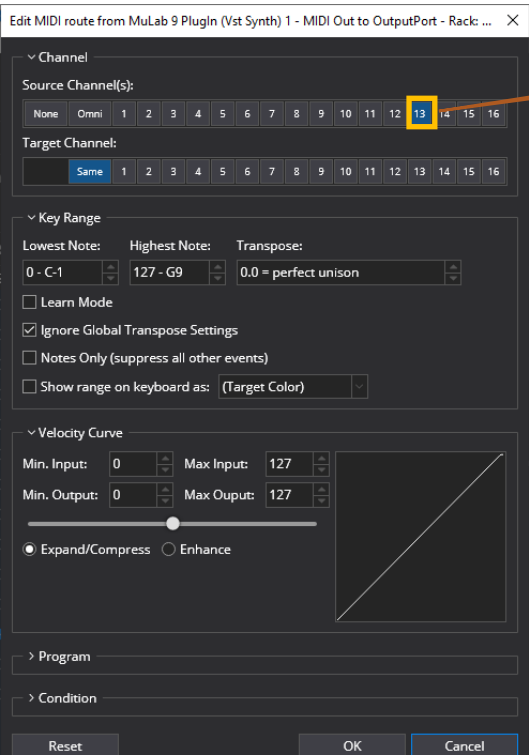
Add these MIDI routes.

For **Output Port – Rack: Main MIDI Out**, click on **Omni** to open the MIDI mapping dialog and set the following, which will filter out channels 13-16 (my control channels). Click **OK** when done.



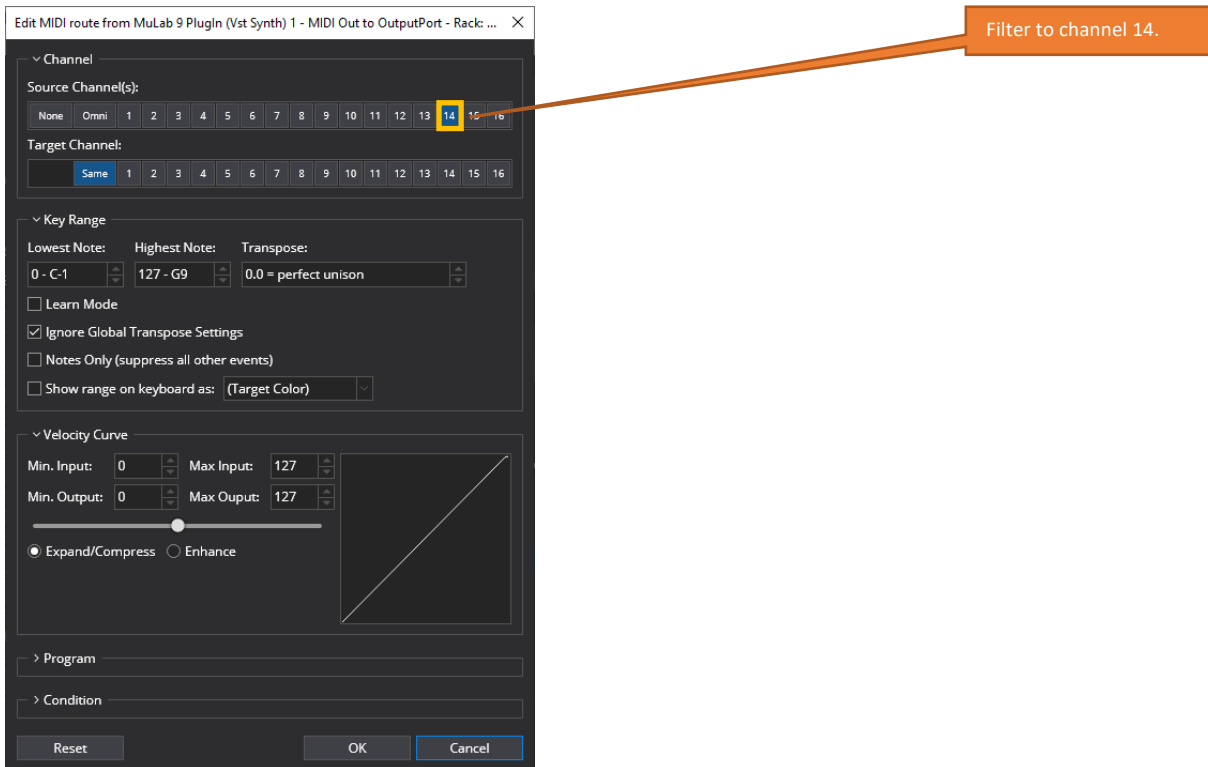
The screenshot shows the 'Edit MIDI route from MuLab 9 Plugin (Vst Synth) 1 - MIDI Out to OutputPort - Rack ...' dialog. Under the 'Channel' section, the 'Source Channel(s)' row has buttons for channels 1 through 12 highlighted in yellow. The 'Target Channel' row has the 'Same' button selected. The 'Key Range' section shows 'Lowest Note: 0 - C-1', 'Highest Note: 127 - G9', and 'Transpose: 0.0 = perfect unison'. The 'Velocity Curve' section has 'Expand/Compress' selected. An orange callout box points to the source channel buttons with the text: 'Filter to channels 1-12 (shift click to select multiple channels)'.

For **Output Port – Rack: Scene Select Out**, click on **Omni** to open the MIDI mapping dialog and set the following, which will only pass channel 13 (my Control channel). Click **OK** when done.

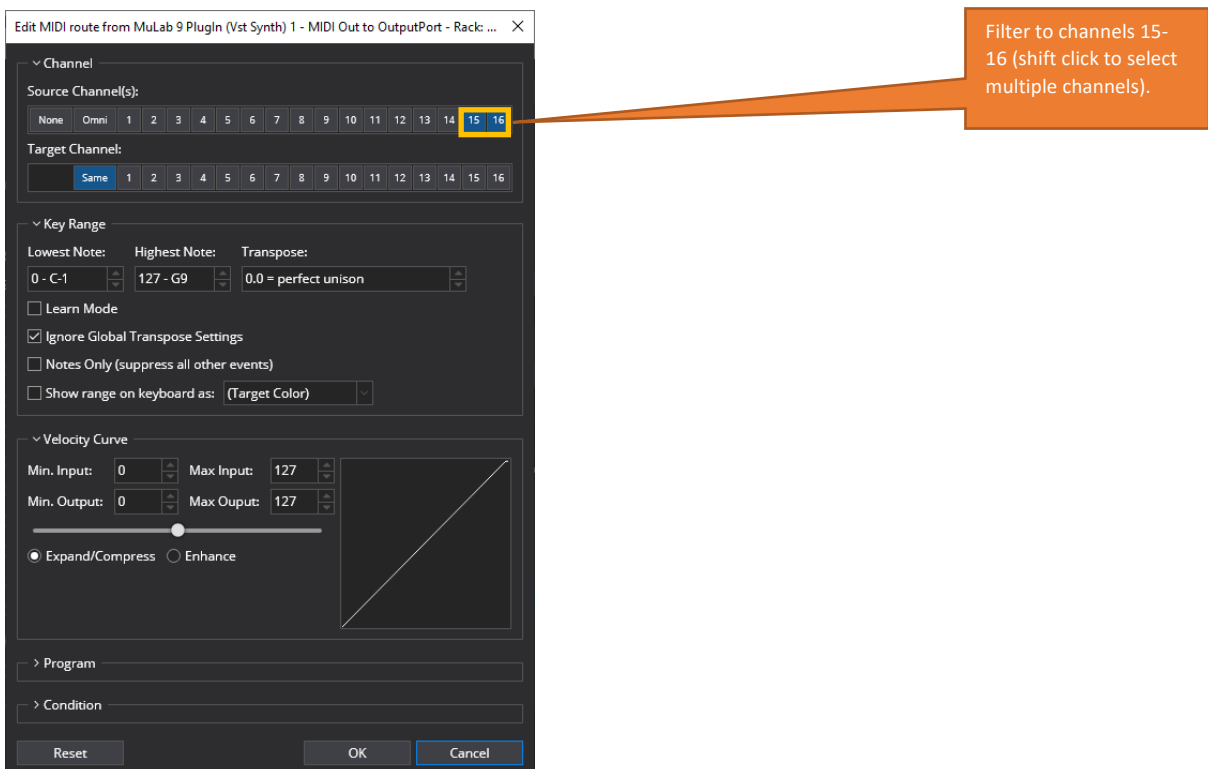


The screenshot shows the 'Edit MIDI route from MuLab 9 Plugin (Vst Synth) 1 - MIDI Out to OutputPort - Rack ...' dialog. Under the 'Channel' section, the 'Source Channel(s)' row has the button for channel 13 highlighted in yellow. The 'Target Channel' row has the 'Same' button selected. The 'Key Range' section shows 'Lowest Note: 0 - C-1', 'Highest Note: 127 - G9', and 'Transpose: 0.0 = perfect unison'. The 'Velocity Curve' section has 'Expand/Compress' selected. An orange callout box points to the channel 13 button with the text: 'Filter to channel 13.'.

For **Output Port – Rack: Video Trigger Out**, click on **Omni** to open the MIDI mapping dialog and set the following, which will only pass channel 14 (my Video control channel). Click **OK** when done.



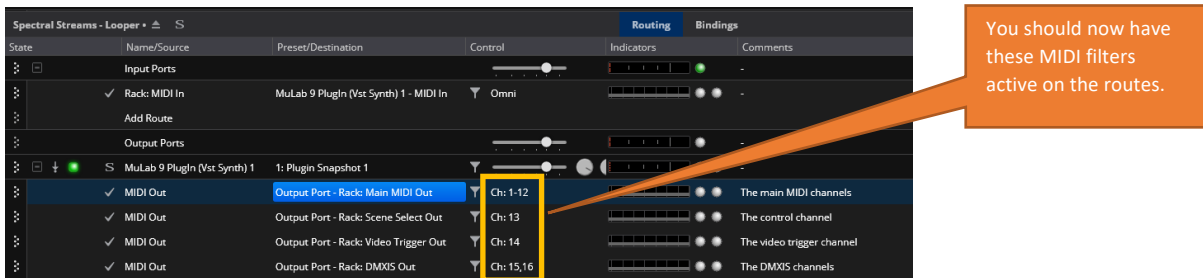
For **Output Port – Rack: DMXIS Out**, click on **Omni** to open the MIDI mapping dialog and set the following, which will only pass channels 15 and 16 (my DXMIS channels). Click **OK** when done.



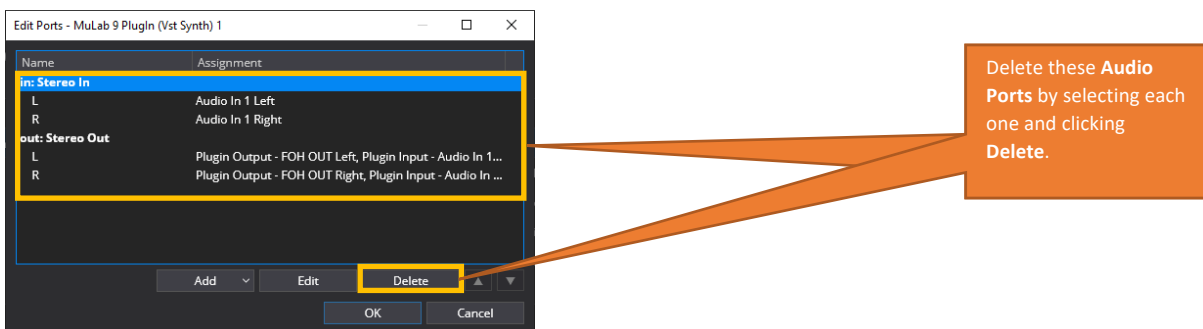
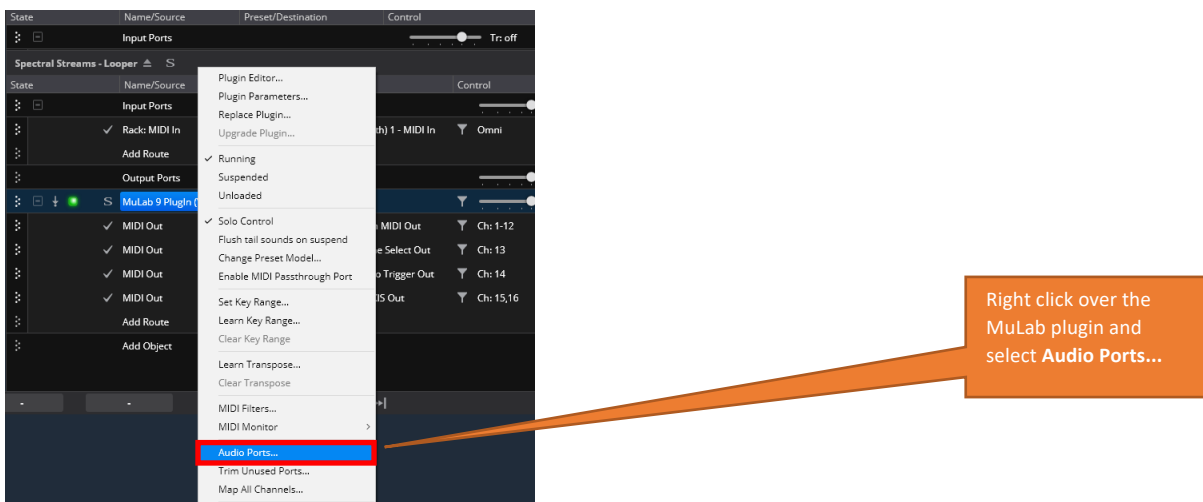
This sets up the MIDI routing and filtering as I need it for my setup, which is as follows:

- CH1 to 12 General MIDI channels that can be allocated as needed from song to song;
- CH13 My general control channel;
- CH14 A channel I use for video cues if I need to trigger video via MIDI control;
- CH15, CH16 MIDI channels for use with DMXIS DMX lighting system to select banks and cues.

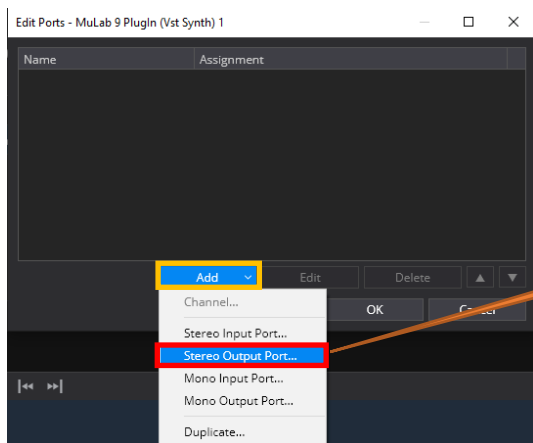
You should now have the following. I have also added some comments to describe the channels.



Next, we need to edit the MuLab plugin Audio Ports, Right click over the plugin and select **Audio Ports...**



Delete the two existing ports, and then select add **Stereo Output Port...**

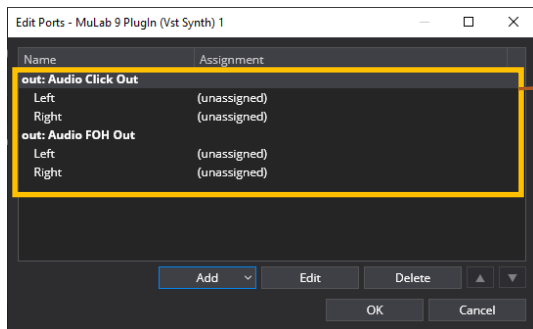


Click **Add** and select **Stereo Output Port**.

Call the port **Audio FOH Out** and click **OK**.

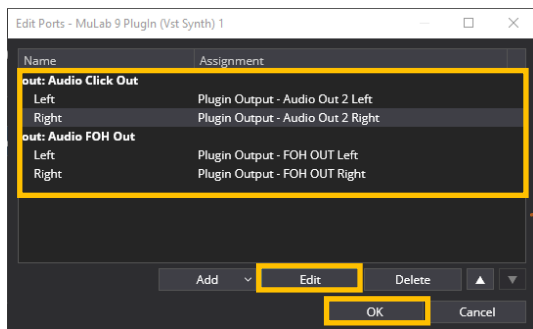
Add another Stereo Output Port called **Audio Click Out**.

You should now have the following ports, which are unassigned.



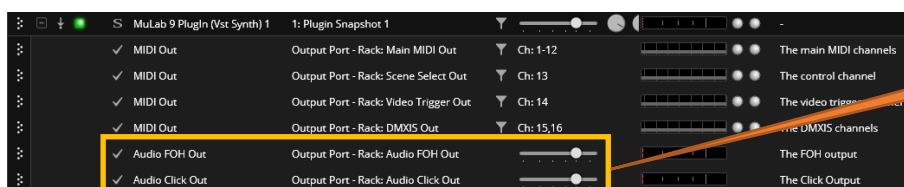
You should now have these unassigned ports.

Click on each sub channel for each port and click **Edit** and set up as follows (note for some reason the Click Output name that we set in the MuLab Modular area is not picked up, so use Audio Out 2).



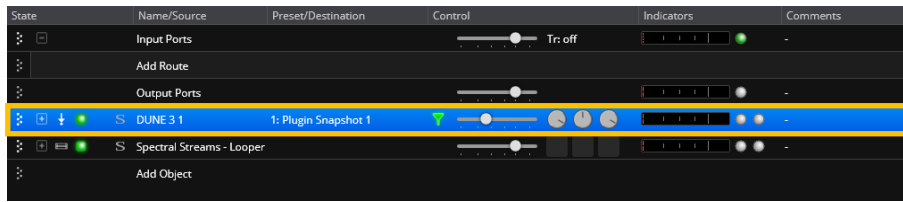
Select each port, click **Edit** and set up the assignments as shown. You should now have these assigned ports. Click **OK**.

Click **OK** and now add the following Audio Routes to the MuLab Plugin.



Now setup these ports.

Now let's add the VST we want to play using the Looper MIDI Data. I am using the Dune 3 VST, which I have set up playing a nice arpeggio that repeats over two bars.

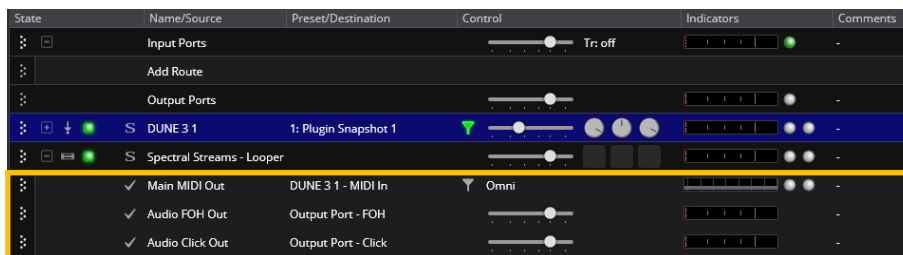


Dune 3 VST set up with an audio route to FOH, and a suitable Arpeggio selected for the looper to play.



Now add the following MIDI and Audio routes to the Looper as shown below, noting that you will need to map to your own audio outputs.

- FOH – The front of house sound that the audience will hear (and I will hear back in my monitoring as a mix with my external instruments);
- Click – The click that I hear in my in-ear monitoring.

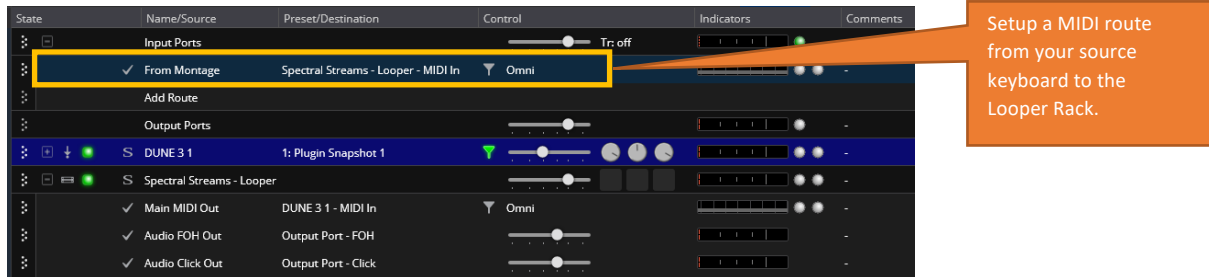


Setup these MIDI and audio routes from the Looper Rack.

At this point you also have the basis for a test song, which you can save for use later on.

Selecting MuLab Scenes from your Keyboard

Remember in MuLab that we set up a **Note Action Map** to trigger the scenes? To get the loops playing it is as simple as setting up an input route from your MIDI keyboard to the Looper Rack. In the example below, I have set up a route from my Yamaha Montage 7 to the MIDI Looper Rack.



Now when you hit C3, C#3, D4 or D#4 (or whatever notes you mapped), then the scenes will play, as expected. With the way we have set up MuLab, the changes only occur on 4 bar boundaries.

If you press the note for a scene to play it, and then press the same note again, then the scene will stop once it has come to the end of the loop.

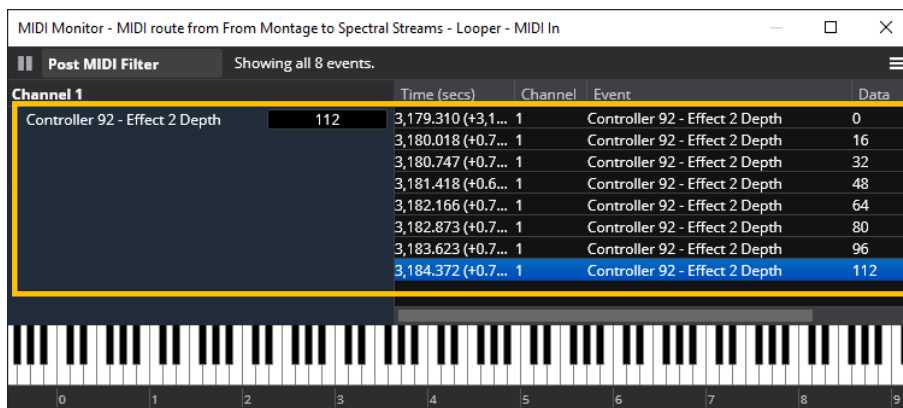
Note also that once you play a scene this way, and because the Cantabile Master Transport is not running, the MuLab Composer transport keeps running. This is important to know, because if you press a note again to trigger a scene, it will only start on the next four bar boundary due to the way we set up the sync. You can of course vary the synchronisation to your needs – there is a lot of flexibility.

Selecting MuLab Scenes from an External Controller

We are now getting into some very specific elements to do with my setup, but hopefully they will provide ideas for yours.

I want to be able to trigger Scenes from my keyboard controller surface buttons, and on my Yamaha Montage 7 I have 8 Scene buttons for scene selection on the Montage. Rather than have yet another control surface, I want to use these buttons⁵.

My Montage is configured to send out CC92 when a scene is selected, and if we look at what happens in Cantabile's MIDI Monitor when I sequentially press the Montage's Scene Buttons 1-8 you can see that each scene has a different CC value, incremented by 16 for each scene⁶.



| Channel 1 | Time (secs) | Channel | Event | Data | |
|--------------------------------|-------------|--------------------|-------|--------------------------------|-----|
| Controller 92 - Effect 2 Depth | 112 | 3,179.310 (+3.1... | 1 | Controller 92 - Effect 2 Depth | 0 |
| | | 3,180.018 (+0.7... | 1 | Controller 92 - Effect 2 Depth | 16 |
| | | 3,180.747 (+0.7... | 1 | Controller 92 - Effect 2 Depth | 32 |
| | | 3,181.418 (+0.6... | 1 | Controller 92 - Effect 2 Depth | 48 |
| | | 3,182.166 (+0.7... | 1 | Controller 92 - Effect 2 Depth | 64 |
| | | 3,182.873 (+0.7... | 1 | Controller 92 - Effect 2 Depth | 80 |
| | | 3,183.623 (+0.7... | 1 | Controller 92 - Effect 2 Depth | 96 |
| | | 3,184.372 (+0.7... | 1 | Controller 92 - Effect 2 Depth | 112 |

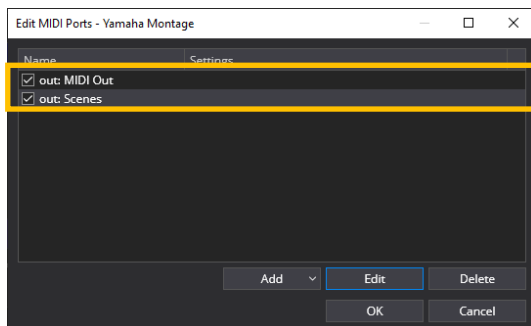
Pressing Montage Scene Select buttons 1-8 generates this CC and values for each button.

I therefore need a way of mapping these CC changes to MuLab's Note Action Map. This is where it gets a little tricky, but, as usual, there is always a way in Cantabile!

I already have a Cantabile Rack for the Yamaha Montage that is listening to the Montage MIDI output on CC92 and mapping those controller changes to program changes: CC92, value 0 is mapped to PC1, CC92, value 16, is mapped to PC2, etc.

This is done by a very simple filter in Cantabile.

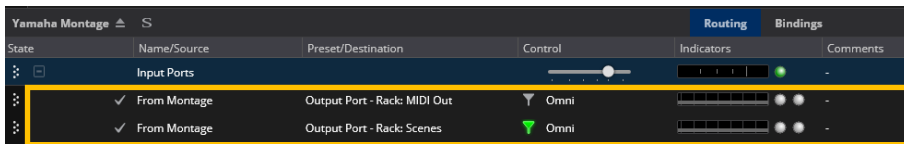
My Montage Rack has two MIDI Output Ports and two corresponding MIDI routes, as shown below



Montage Rack MIDI Ports and Routes. Note that the Scenes port has an active MIDI filter.

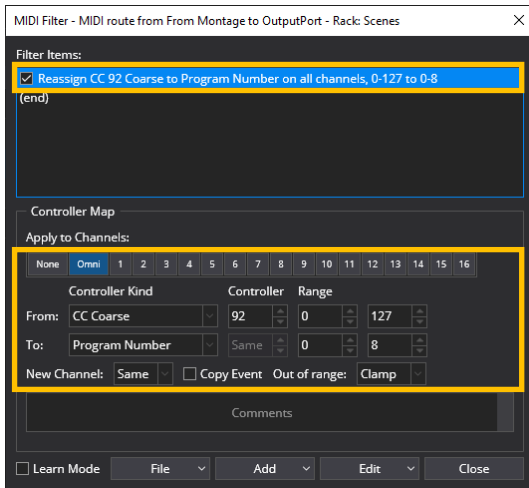
⁵ It can be a little more complicated than that if I also involve my FC300 MIDI foot pedal for when I have both hands on the keyboards, but that would over complicate the description!

⁶ The 128 possible CC values have been divided by 8.



The first route is simply a “through route” that passes all MIDI data from the Montage onwards to the Rack’s MIDI Out Port.

The second route goes to the Rack’s **Scenes** port, which has the following MIDI filter, that is remapping the CC92 values to Program numbers.



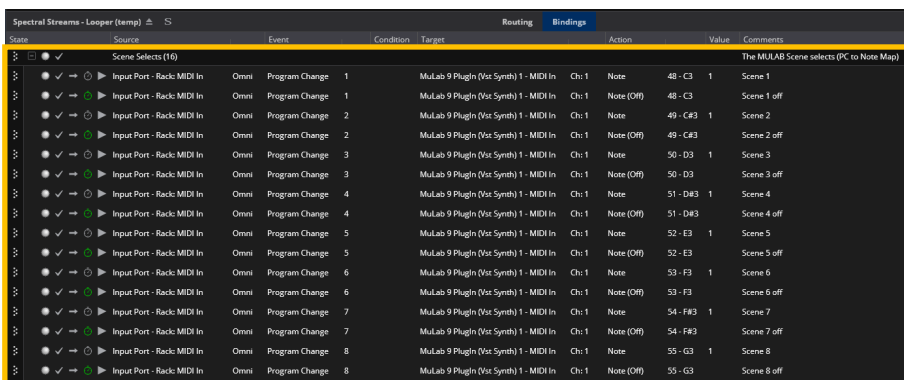
Montage Rack Scenes MIDI Port Filter that maps CC92 values to program changes 1-8.

Back to my Montage Rack at song level, I have the following route setup to connect the Montage Rack’s MIDI Scenes output to the looper rack.



Montage Rack Scenes MIDI Port is routed to the Looper Rack MIDI input.

This gets the PC changes going to the Looper Rack, and I need to map those Program Changes to the MuLab Note Action Map that we set up earlier, which is achieved via the following bindings (a note on and a delayed note off for each program change). I set the delay for the Note Offs to 100ms.

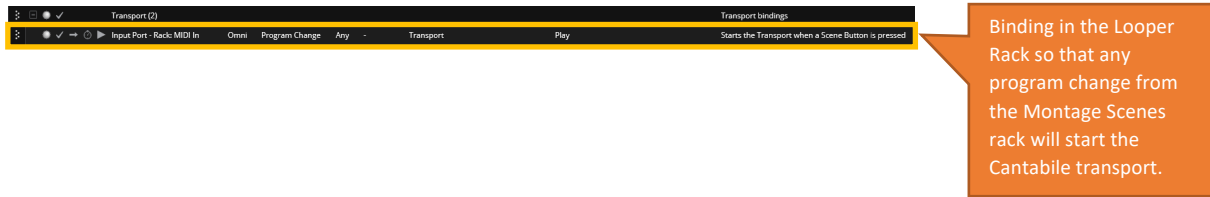


These bindings in the Looper Rack translate the program changes output from the Montage Scenes Rack to the Note values in the MuLab Note Map.

The duplication in the bindings for each program changes, means that MuLab sees both a note on and a note off, so there is no chance of stuck notes.

It would be nice if MuLab can trigger scenes in more direct way from the CC, but if it can then I have not found that yet. This works for now.

And I also have the following Transport related binding, which starts the Cantabile Master Transport when any program change is detected (e.g., I have pressed a Scene Button on my Montage).



Because MuLab is synced to the Cantabile Master transport, then it also starts as well.

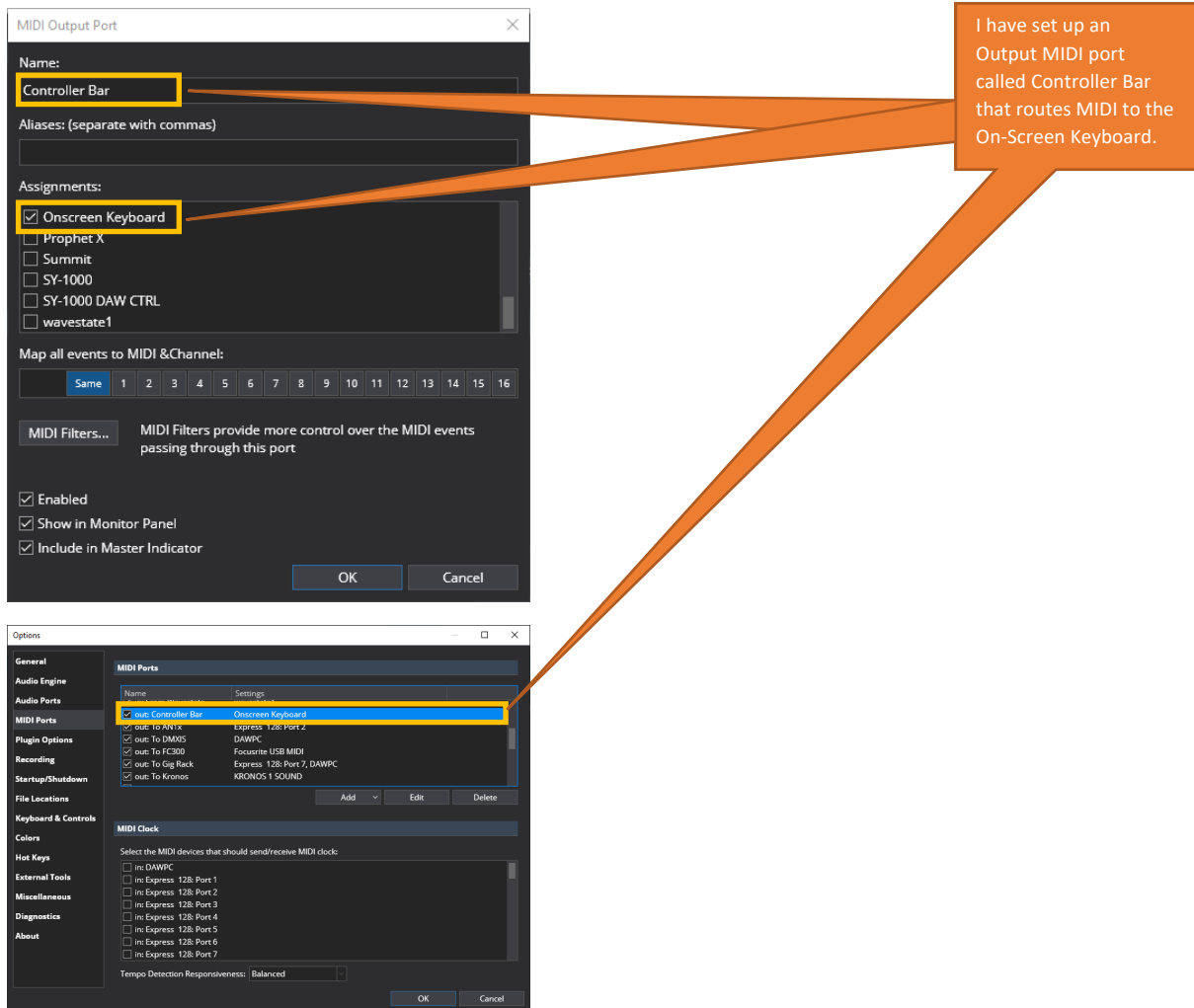
I do this as I can then see the tempo indicators in Cantabile (I will also hear the click of course), and when I stop the Cantabile transport, then the MuLab Composer transport (which is started when you play any scene) is also stopped.

Showing MuLab Scene Changes in the Cantabile Window

I wanted a way of showing when a scene change request is actually activated. When I press a Scene button on my Montage the feedback on the Montage is instant, but with the way I have setup MuLab in the examples, the Scene will only change on four bar boundaries, not instantly. I want a way to see when a **Commanded** scene change actually occurs.

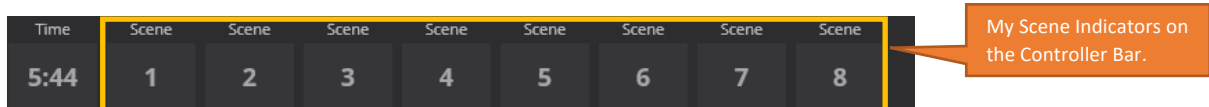
This section describes how I have achieved this using the Cantabile **On-Screen Keyboard** and the **Controller Bar**⁷.

Under **Tools/Options/MIDI** I have set up a MIDI output port as follows.



In the main Cantabile window, ensure that the **Controller Bar** is visible (View/Controller Bar menu option).

I have set up the Controller Bar to have 8 Scene Buttons as shown below, and the buttons will act as Scene indicators.

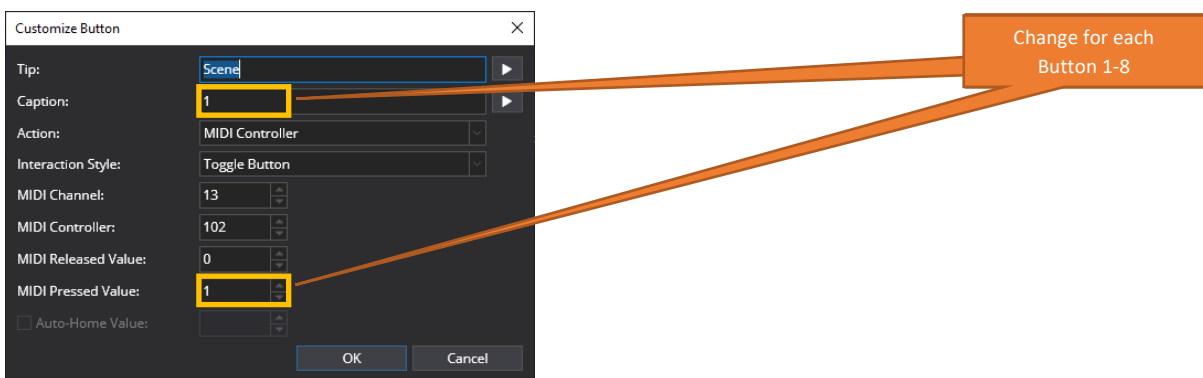


⁷ With thanks to Dave Dore for his help on how to use the controller bar in this manner.

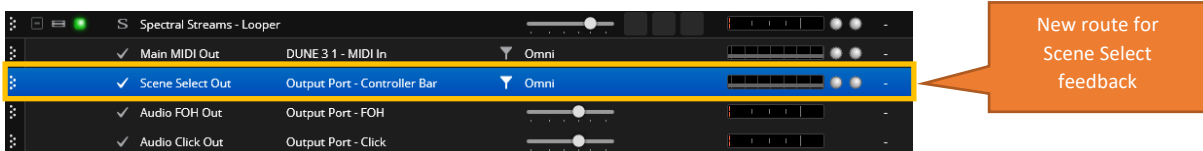
To add the buttons, right click over the Controller Bar and select **Add Button...**



For each button, set up as follows, where the Caption and MIDI Pressed Value parameters are incremented from 1 to 8 for each button.

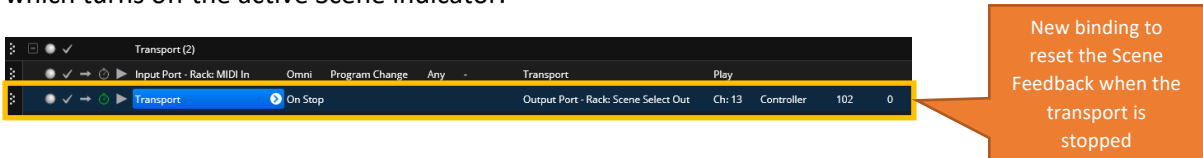


Now back in the Looper Rack MIDI Routing at Song level, add the following route.



This will send the Scene Select data (CC102 on Channel 13) that is contained within the MIDI data in each MIDI clip to the Controller Bar, and it will light the appropriate button in the Controller Bar when the scene changes.

I also add a binding in the Looper Rack Bindings to send CC102, value 0 when the transport stops, which turns off the active Scene indicator.



Creating Linked Rack States to select Different MuLab Projects

The benefit of having MuLab setup within a Cantabile Linked Rack is that you can drop the same rack into several songs in your set and use **Rack States** to select the MuLab Program to be used by each song.

First of all, you need a number of different MuLab projects, and in my testing I have created the following projects, which use the same MuLab project structure as described, but with different Composer configurations (I have been describing how I set up Loop Test 2 in the above):

- **Loop Test Default** (8 Scenes, MIDI and Click);
- **Loop Test 1** (2 scenes, MIDI, Taurus, Stylus RMX and Click);
- **Loop Test 2** (4 scenes, MIDI, Taurus, Stylus RMX and Click).

Which is enough to show how using Rack States to recall a specific MuLab project will work.

The three projects are shown below for comparison.

Loop Test Default

The first test I ever did, which has 8 scenes (testing all eight scenes on my Montage) with a single MIDI track and a single click track).



Loop Test 1

First test of adding more audio tracks, limited to two Scenes.



Loop Test 2

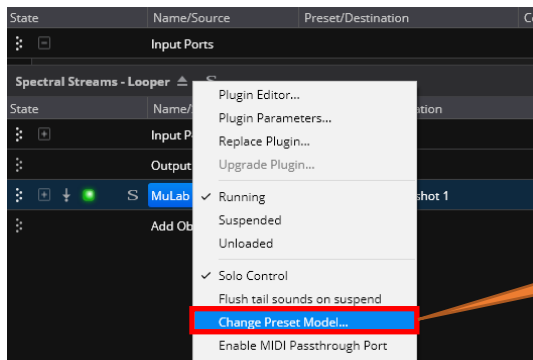
The one we have just described how to setup.



Now we have these 3 MuLab projects, we will set up 3 Looper Rack states, where each state within the rack recalls one of the projects.

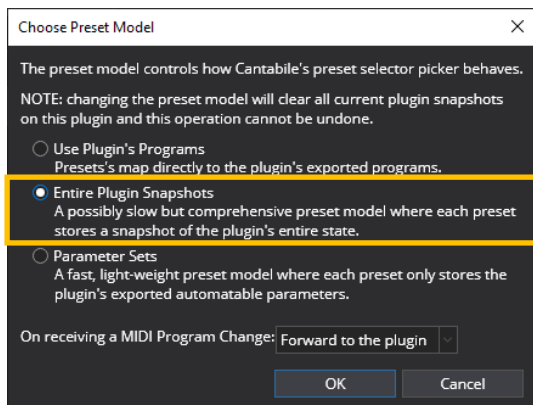
In Cantabile double click on the Looper Rack to open it, and the first thing we are going to do is ensure that the right **Preset Model** is selected.

Right click over the MuLab object and select **Change Preset Model...**



Right click over the MuLab plugin and select **Change Preset Model...**

Ensure that **Entire Plugin Snapshots** is selected and click **OK**.

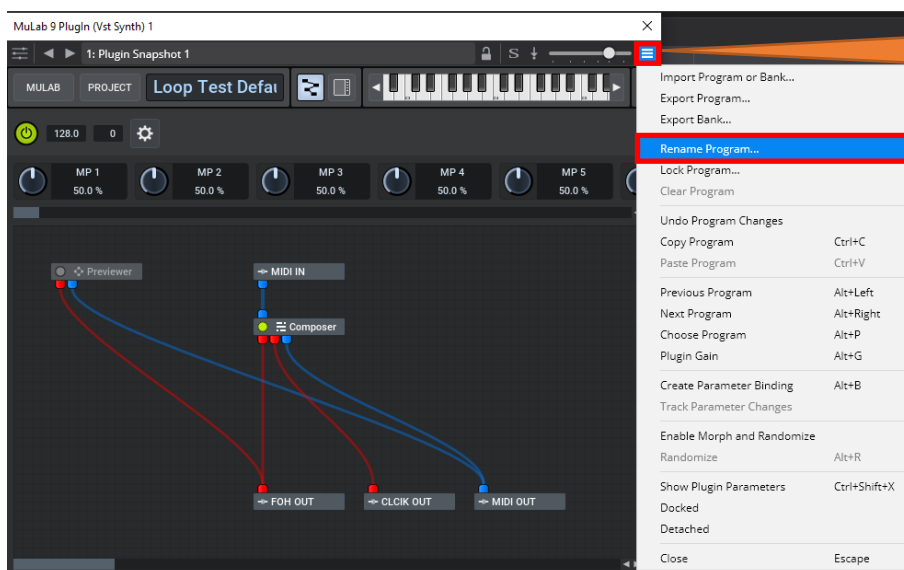


This is the preset model we want so that the entire MuLab Project is stored within a Cantabile Rack State.

This ensures that the entire plugin state is stored in each State that we create.

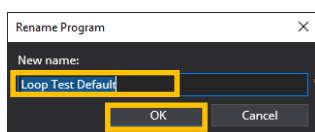
Open MuLab and ensure that the Loop Test Default Project is open.

Click on the top right “Hamburger” icon, and select **Rename Program...**



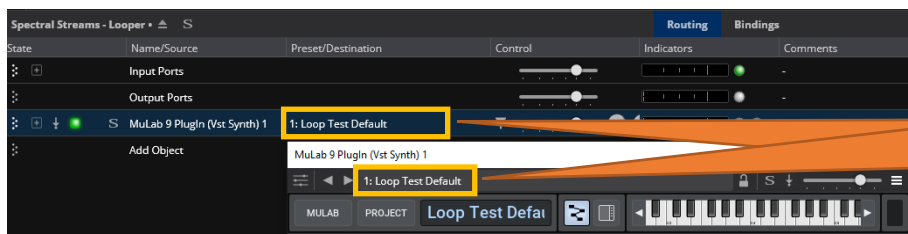
Click here and then select **Rename Program ...**

Enter the name of the MuLab Project and Click **OK**.



Rename the preset (program name) and click OK.

You can see that the preset name in the Plugin in Window and the Snapshot state has changed.



The name of the Program is shown as the Preset name.

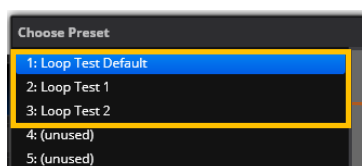
Click on the Preset and select **2: Unused** and you will now get the following. Note that in this rack state, you once again have a blank MuLab project.



In MuLab, open your second project, in my case **Loop Test 1**, rename the Program to **Loop Test 1**.

Back in Cantabile repeat the above for Plugin Snapshot **3: Unused**, load **Loop Test 2** in MuLab and rename the program to **Loop Test 2**.

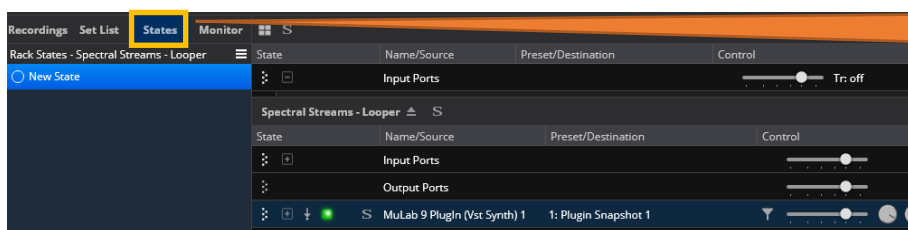
Now when you click on the Preset Button, you see the following three programs in the Menu (and the remaining snapshots are empty).



These are our defined Presets. All other presets are empty.

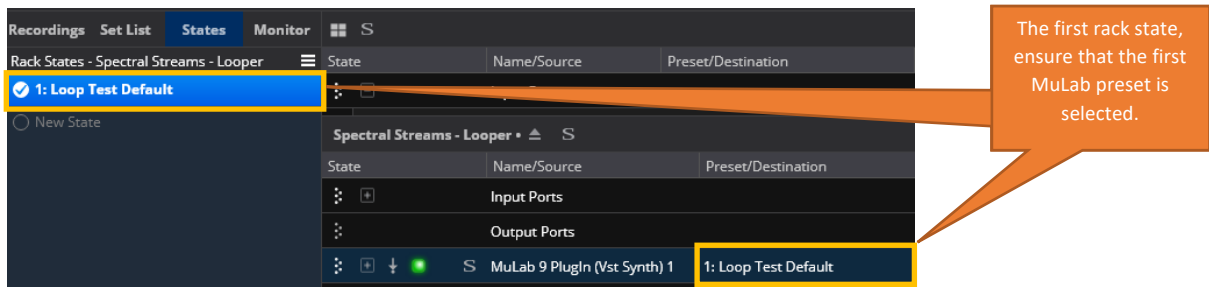
Now we can create the states for each program.

Whilst the Looper Rack is still open, select **States** in the left-hand Pane.

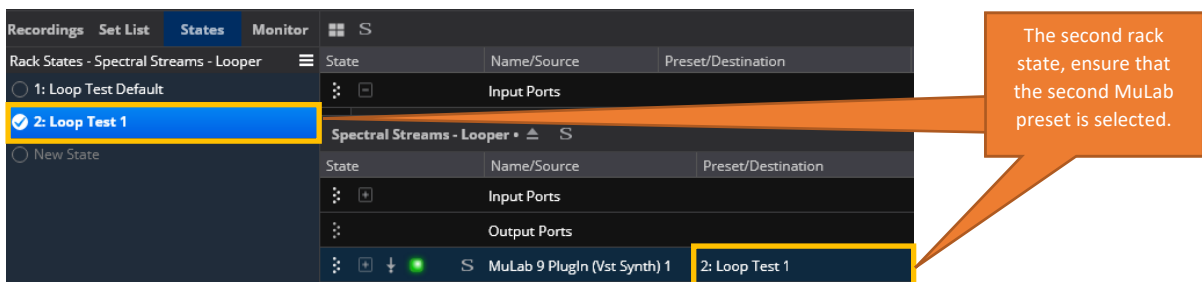


Click on States so we can add the states to the Rack.

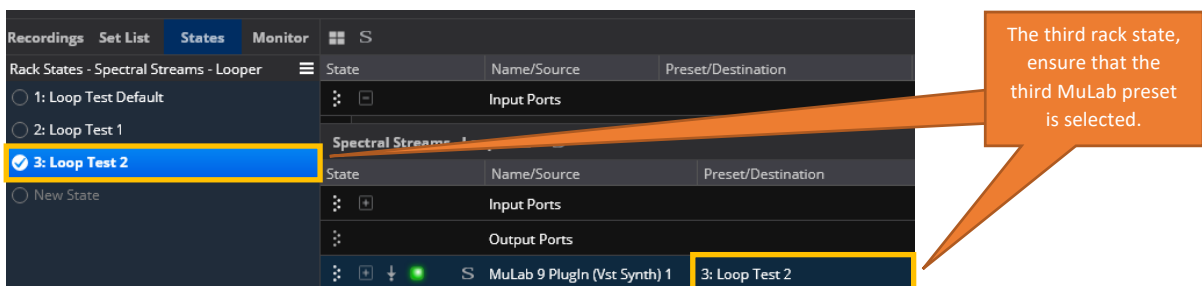
Click on **New State**. Type in the name **Loop Test Default** for the State, click **OK**. Ensure that the selected preset is **Loop Test Default**.



Click on **New State**. Type in the name **Loop Test 1** for the State, click **OK**. Ensure that the selected preset is **Loop Test 1**.



Click on **New State**. Type in the name **Loop Test 2** for the State, click **OK**. Ensure that the selected preset is **Loop Test 2**.



When select each state, you should see the preset for the MuLab plugin changing.

Now is a good time to save the Looper Rack again.

Now set up two songs in Cantabile that are identical to what we have created above. Call them **Loop Test - Song 1** and **Loop Test – Song 2**.

In **Loop Test - Song 1**, set the Loop Rack State to **2: Loop Test 1**.

The screenshot shows the Cantabile interface with the 'Set List' panel on the left and the 'States' panel on the right. In the 'Set List' panel, '19: Loop Test - Song 1' is selected. In the 'States' panel, the 'Spectral Streams - Looper' rack state is highlighted, and its 'Preset/Destination' column shows '2: Loop Test 1'. An orange callout box points to the '2: Loop Test 1' preset with the text: 'For Test Song 1, select the 2: Loop Test 1 Rack State. (Because this is a Rack, the Preset column shows the selected Rack State).'

In **Loop Test - Song 2**, set the Loop Rack State to **3: Loop Test 2**.

The screenshot shows the Cantabile interface with the 'Set List' panel on the left and the 'States' panel on the right. In the 'Set List' panel, '18: Loop Test - Song 2' is selected. In the 'States' panel, the 'Spectral Streams - Looper' rack state is highlighted, and its 'Preset/Destination' column shows '3: Loop Test 2'. An orange callout box points to the '3: Loop Test 2' preset with the text: 'For Test Song 2, select the 3: Loop Test 2 Rack State. (Because this is a Rack, the Preset column shows the selected Rack State).'

In effect the Rack States are giving you presets for each Song, and with the Looper Rack being a linked rack across the songs, then all the MuLab projects within the Rack will be loaded as part of the set list pre-load.

Conclusion

Hopefully this guide has been useful and shows how to setup a Looper within a Cantabile environment.

When I started writing this guide, I did not envisage it being as large as it ended up at over 50 pages, but plenty of hopefully helpful screenshots always makes for a large document!

Also, the value of writing things down soon became apparent when I created the setup from scratch again for the guide, as it took me time to remember some of the smaller steps that were needed, and that was just after a week away from it! MuLab is very powerful, but it can take a little while to find your way around, and there are lots of little settings to remember.

If nothing else, this guide will be invaluable for me in a few weeks' time, let alone a few months or years! And I hope it has value for other Cantabile users who would like to have a looping solution within Cantabile.

Derek Cook