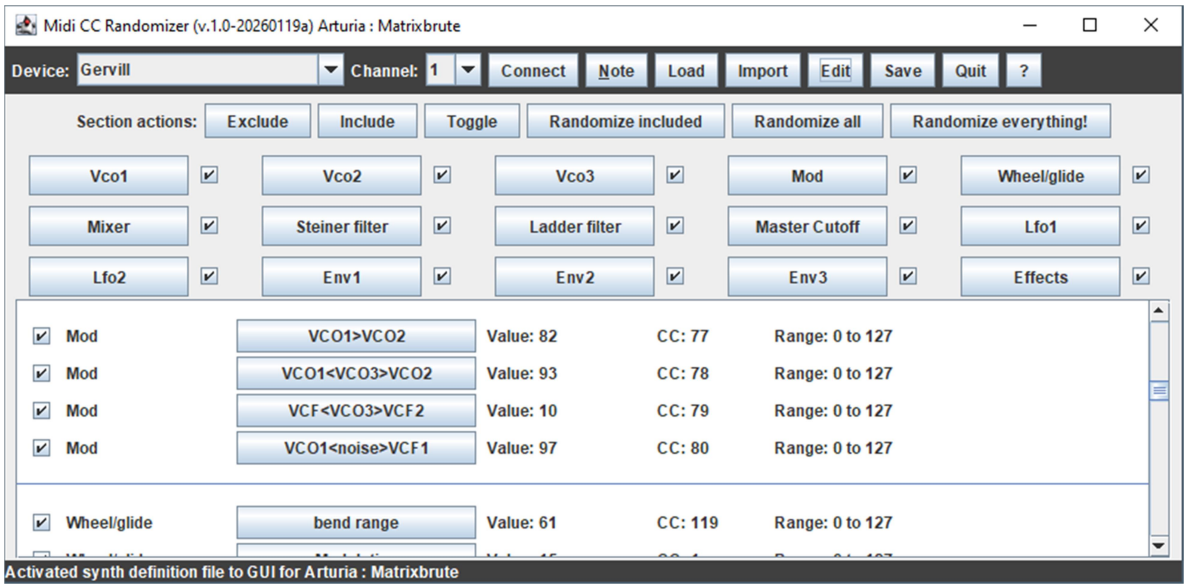
MIDI-CC-Randomizer (MCR)

The Generic MIDI CC Randomizer

v. 1.0-20260119a



Important notice:

- This software is written for **my own use**.
- I'm **not** associated with Arturia (or any other company) in any way.
- This is **not** to be considered as **professionally released software.**
  - Suggestions welcome, implementing them if I feel like it ☺
- N**o guarantee in any way** is given. **Use at your own risk.**
- This is shared for free in the hope that it can be useful for others.
- You're **not allowed to sell** this software, or charge anything for it.
  - This includes that you include it in a package you sell.

If you feel this software has a value to you,
please donate any amount to a local charity ☺

Github: https://github.com/MatrixbrutePatchViewer

Contact: matrixbrutepv@hotmail.com

Contents

The Generic MIDI CC Randomizer (MIDI-CC_RND or MCR for short)

This is a java application that should run on any platform (developed and tested on Windows only). It requires java 11 (or later).

This application was originally created to randomize any parameter on the Arturia Matrixbrute that can be changed with MIDI CC. It is written in a generic way, which means it is easy to more MIDI equipment accepting midi CC by creating custom **synth definition files** that can be loaded.

The synth definition file can also be adjusted to limit the number of parameters available for a given synth, or to change the range of any parameter (f.ex from 0-127 to 20-35 if that is a range that makes more sense for a given parameter – probably envelope attack times of 100+ is not so interesting...). You can also create different definition files for the same synth, if you want one that covers all parameters, another that groups different parameters together for randomizing with one click etc.

It is also possible to import definition files from the **Midi Guide** github site (MIDI CC & NRPN database), which has an ever expanding range of Midi CC definitions. See https://midi.guide/ and https://github.com/pencilresearch/midi for more information, and to download definition files to import into MCR.

The minimum you need to know to enjoy MCR!

To use MCR in the best possible way, you should know this (for more information on each topic, check the rest of this manual!):

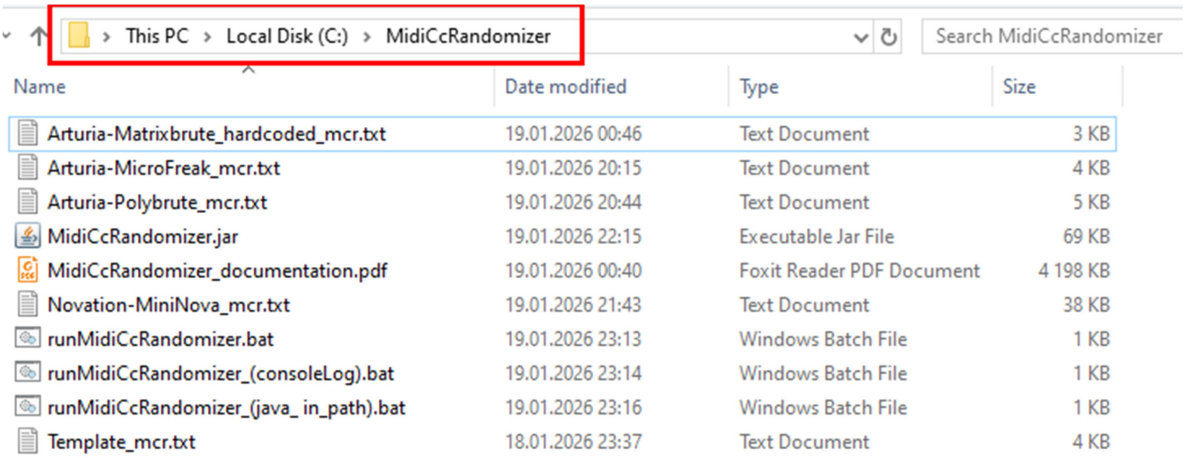
- You can import an existing synth from the Midi Guide project!
  - No wasted time typing it in yourself
  - You might need to adjust the imported data to fit your preferences.
- You can change the value of any parameter with click + drag up / down on then **Value** field
  - Use this to fine tune a parameter (or as a short cut to set parameters that are hidden in menus).
  - Lock it afterwards by unchecking the parameter include checkbox if you found the sweet spot!
- You can always edit a loaded synth to make it better for you!
  - Change the appearance, like window or component sizes, parameter range and grouping etc.
  - You might have other preferences on what to randomize, f.ex will randomizing tuning mostly make bad results!

How to install - Windows

This chapter shows how to install MCR to the folder **C:\MidiCcRandomizer** and having **Temurin java 25** installed into the same folder. If you want to install to somewhere else, adjust paths accordingly. This application is written in java for java version 11. To run it you need java to be available on your machine in version 11 or higher.

The installation as shown here will be an installation that will not affect anything else on your PC, and it should work without any further changes (recommended if you just want it to work, quickly). The folder can also be moved after installation.

1: Download the **MidiCcRandomizer.zip**, and then unpack it to get this folder structure on C:\



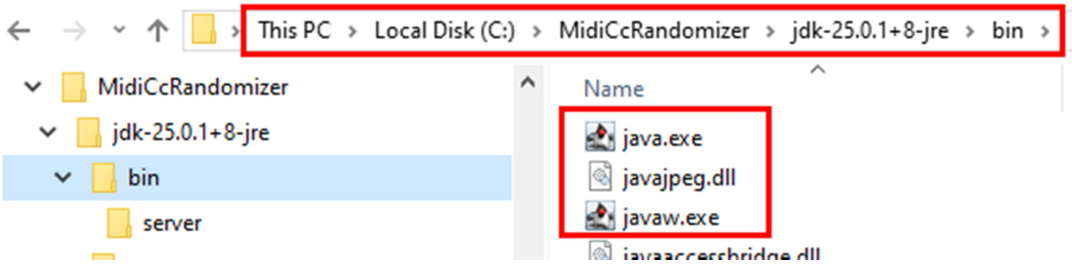
C:\MidiCcRandomizer will then contain all files needed to run MCR.

What to do next depends on your setup regarding java. If you have java 11+ installed, and in your path you are finished and can run the application by double-clicking **runMidiCcRandomizer_(java_ in_path).bat**. Try it now, if it works, your good – if not continue below:

If you need to install java, it can be done in a way that will not affect anything else:

2: Download https://adoptium.net/temurin/releases/?version=25 (or any version from 11 or higher)
You will then get a file like this: **OpenJDK25U-jre_x64_windows_hotspot_25.0.1_8.zip**

3: Unpack it into **C:\MidiCcRandomizer** so you get the following folder structure (just one directory should be added to C:\MidiCcRandomizer, which is **jdk-25.0.1+8-jre** (or similar named):



4: Then rename the folder **jdk-25.0.1+8-jre** (it might have a different name) to **java**. This is what the bat-files expect. If you like you can change the path in the bat to match your java-folder instead.

5: Then start the application by double-clicking on **runMidiCcRandomizer.bat** (you can edit that file if you want to load another synth as default)

How to install - Mac (not tested)

Everything is written in java and should also work on Mac, but it's not tested. As for windows, it needs java 11+ and starting it should be similar as on windows (but with "mac"-specific paths etc). Script to start might be something like this (but java might need a full path prefix if the needed version is not in the path):

java -jar MidiCcRandomizer.jar

You can also try the same way as for windows installation, unpack **MidiCcRandomizer.zip** to a location on disk. Then download the mac-version of java as a zip. Unzip it into the same folder and rename it to java/. See pictures in previous chapter! When installed that way you should be able to run it from cmd-line / terminal like this:

cd MidiCcrandomizer
./java/bin/java –jar MidiCcRandomizer.jar

Update MCR

If you are **upgrading** to a newer version of **MidiCcRandomizer**, you can make a backup of the files in the install folder if you wish (except the folder **java**/ which will never be changed by MidiCcRandomizer).

You can then unzip into the folder, but be sure to replace any files that already exist there. If you have changed any of the included definition files (*_mcr.txt) you must make a backup of those, or not unpack those files.

How to run

To run the application 3 bat-files is provided for windows. Two of those bat's assume java to be installed in a folder beside the location of the bat file (C:\MidiCcRandomizer\) called java\ so the full path to the java executable will be **C:\MidiCcRandomizer\java\bin\java.exe** as described in the install chapter above.

Run **runMidiCcRandomizer.bat** to start the app the normal way.
Run **runMidiCcRandomizer_(consoleLog).bat** if you need to see log-data in the console while running.
Run **runMidiCcRandomizer_(java_ in_path).bat** if java is already installed on your system.

You can also run it manually from the cmd line (assuming java is in the path, if not use **java\bin\java -jar**):

java -jar MidiCcRandomizer.jar	Start with Arturia Matrixbrute
java -jar MidiCcRandomizer.jar MySynth.txt	Start by loading the synth MySynth.txt (must exists!)
java -jar MidiCcRandomizer.jar - help	Show help:

NOLOGTODISC: No logfile log_mccrnd.txt will be created.
LOGTOCONSOLE: Log will be written to the console.
<filename>: Will try to load this MCR synth definition file when starting.
If no file is loaded, the default Artura Matrixbrute is loaded.
HELP: Show this help.

The default is to log to file (turn off with **NOLOGTODISC**), and not to log to console (turn on with **LOGTOCONSOLE**).

You can combine all those also, in any order. Anything not recognized is assumed to be a file name:

java -jar MidiCcRandomizer.jar NOLOGTODISC LOGTOCONSOLE MySynth.txt

The log is reset every time you start the application, and one backup is kept so no worry about disc space. If you run multiple instances at the same time they will share the log file, but each instance starting will move the current log file to backup (meaning logs can be confusing when running more than 1 at the same time).

Using MCR

When starting the program the hardcoded definition for **Arturia Matrixbrute** is loaded. This can be changed by starting the program with an **argument** pointing to the synth definition file to be loaded as default. Start the app with: Java -jar MidiCcRandomizer.jar **YourDefaultSynth.csv**

Most buttons in the GUI have a tooltip, so to see info about any button just hold the pointer still over it.

Connect to your synth

When started, you must **connect** to your synth with the correct **Device** and **Channel**. Select your values, and click **Connect**. Click "**Test note**" to send a C4-note to verify your connection. **Note**: Gervill is java's default midi device, and will probably show up as the default. To connect to VST's in a DAW you need a loopback device – see the chapter ***Randomizing VST's in a DAW (or a stand-alone VST).***

If the loaded synth does not match your connected synth (see manufacturer and model in the title bar), you must use "**Load…**" (for mcr .txt-files) or "**Import…**" (for midi guide .csv-files) to be able to use MCR. See other chapters for descriptions of these functions.

The common section

There main window contains a **common section** that applies to every synth (the two rows at the top):

This can be used to change which sections to include/exclude (indicated by the check box next to each section), and to randomize some or all of the parameters:

- **Exclude all / Include all:** Unchecks and checks all sections to quickly select none or all sections.
- **Toggle:** All checked is unchecked, and all unchecked and checked.
- **Randomize included**: This will randomize the included sections, and only randomize the included parameters for those sections.
- **Randomize all**: This will randomize all sections (ignoring the include checkbox on sections), but only included parameters are randomized
- **Randomize everything**: Ignores all "included" checkboxes, randomizes every parameter in every section.

Note: If the window is too narrow, the section checkboxes might disappear – just widen the window to fix!

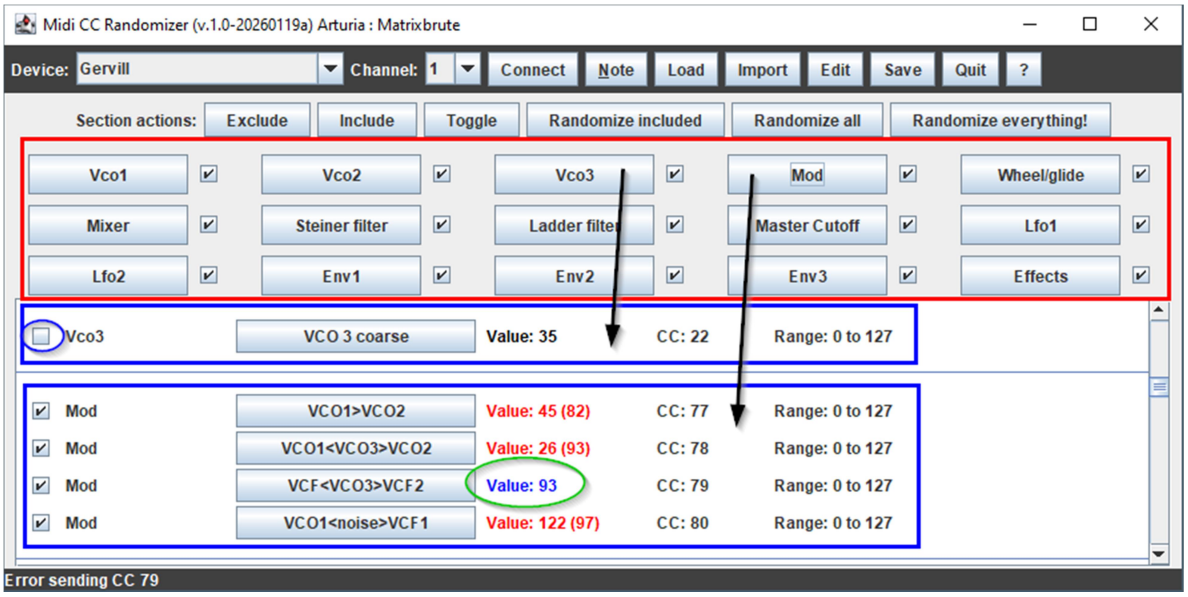
The synth section

There is also **a synth section**, representing the sections and parameters in the currently loaded synth. It's located below the **Section action** buttons.

The **sections** are located at the top of the window (marked in a red square). A **section** represents a group of related parameters, like LFO1 or Envelope 1.

Each section have one button and one checkbox (**include**). When **include** is checked, the section will be randomized by **randomize included** or **randomize all**.

The list of parameters for each section is in the scrollable lower part of the window. The checkbox on the left side is used to include this parameter when randomizing. **Note**: The **include** checkbox for **section** and **parameters** is ignored when using the button **Randomize everything!**.



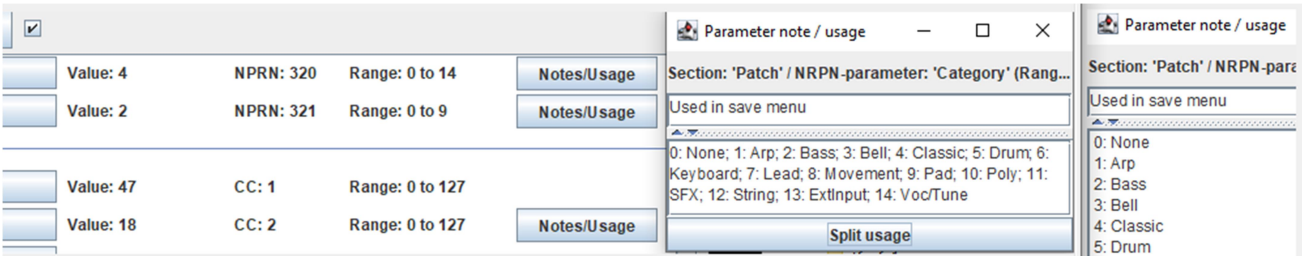
Clicking on section button Vco3 will not have any effect, as it is one parameter that is unchecked (blue circle).

The image above shows MCR after clicking the section button **Mod**, and then click+drag over the **value** for **CVF <CVO3> VCF2** parameter. All the red values were randomized by the last operation (= clicking **Mod**) – showing the previous value in (). The blue **value: 93** (in the green circle) was changed manually after randomization by clicking (and holding) on the value, then dragging the mouse up/down to change the value. Midi CC is sent while dragging, so you should be able hear the effect if a sound is playing when you drag. This is useful to set one parameter to a fixed value before randomizing further. Remember to uncheck the **include** checkbox on the left side when you are happy with your set value.

Tips: Click+drag can also be useful to quickly control all CC/NRPN parameters on your synth, especially if it has a lot of parameters hidden in deep menu trees!

Notes & usage information

If the parameter have notes or usage information, a button **Notes/Usage** is shown that will open a separate window showing this information. It is also presented as a tooltip, which works fine if it's short! The note is shown in the upper text field, and usage in the lower. Click **Split usage** to format the usage by splitting on ";" (rightmost window). This is standard notation used in Midi Guide.



Note: Hover the pointer over any NRPN (f.ex "**NRPN: 321**" in the picture above) to get a tool tip showing the MSB / LSB value representing 320. This only applies to NRPN, as CC is max 127.

MIDI CC, and NRPN

MCR supports both ordinary CC and NRPN parameters, but you can define only one of them for a given parameter. It is of course possible to define 2 separate parameters for the same thing, where one is CC and the other is NRPN – but it will probably not make sense to randomize both at the same time!

MIDI CC

MIDI CC is defined in the MIDI protocol as a standard MIDI message, where you send the CC number, and the value. Both are 7bit (0-127). Many parameters are defined in the MIDI standard, and works across different synths from different manufactures. The standard also allow for 32 CC-commands that can use 14 bit values, by sending a CC (0-31) for MSB, then a CC 32 higher for LSB – this is not (yet?) implemented in MCR.

NRPN

NRPN means **Non-Registered Parameter Numbers**, and are vendor specific "CC" messages that also can be high resolution. Both the message type and the value are 14bit (0-16383) which allows greater resolution of a parameter.

Technically a **NRPN** message is sent using four standard MIDI CC messages:

- **CC 99** (Parameter Number MSB) & **CC 98** (Parameter Number LSB) define which parameter.
- **CC 6** (Data Entry MSB) & **CC 38** (Data Entry LSB) define the value of that parameter.

It is believed that most (or all?) manufacturers follow this standard, and send / accepts NRPN with CC 99+98 + 6+38 in that order. In case your synth requires a different order for these CC messages, you can set that using the tag **NRPN_ORDER** in the synth definition file. You should be able to observe the order by capturing midi from your synth and looking at the order of CC's mentioned above. More details in the chapter describing the definition file format.

There is also mention of a "best practice" to terminate the NRPN parameter change by sending a "null NRPN", by sending **CC 101** value **127** + **CC 100** value **127**. This is not done by default in MCR, but can be configured by including the tag **NRPN_TERMINATE** in the synth definition file.

Some synths (f.ex Novation Mininova) might have issues when receiving parameter value MSB, when it's 0. To turn off sending CC 6:0 use **NRPN_ONLY_LS**B in the definition file.

This info in this chapter is only relevant for NRPN-messages.

Creating a custom synth definition file

A **synth definition file** is a simple text file containing tags (for controlling certain things, like size of windows, sections etc) and all the parameters. The sectionsis derived from the parameter list (or by using tags). It is also possible to include comments. Everything after the first # on each line is ignored.

The parameter data is comma-separated, which means you can't use , in the actual data without putting the whole field inside double-quotes "".

Please look at the file **Template_mcr.txt** for more details about the synth parameter file format. You can also import a csv-file from **Midi Guide** and edit then save that as a MCR file. It will contain commented lines with all of the tags that you can enable if needed. Hopefully a great start to get into the file format!

You can create a new synth definition file in multiple ways:

- Load an existing MCR file (or use the default matrix brute one), edit it, and save it.
    - This is useful for making adjustments to an existing synth.
- Import a Midi Guide file (and optionally edit it), and save it as a MCR file.
    - This is when Midi Guide already has a definition for your synth. You might probably need to do some editing also!
- Type it in from scratch in notepad (or any text editor) – based on the **Template_mcr.txt**
    - This is the normal way for a synth that is not found in Midi Guide.
    - You should actually type it in Midi guide format, and upload it to that the Midi Guide project, then import it! Keeps everybody happy ☺

Creating a synth definition file from scratch

You can create a synth parameter file from scratch, or by copying the included **Template_mcr.txt**. This file will include all details of the format, and the possible **tags** you can use for controlling the layout and data. It is also commented.

In general there are 3 types of lines - line separator can be **\n** (mac/unix/linux) or **\r\n** (windows):

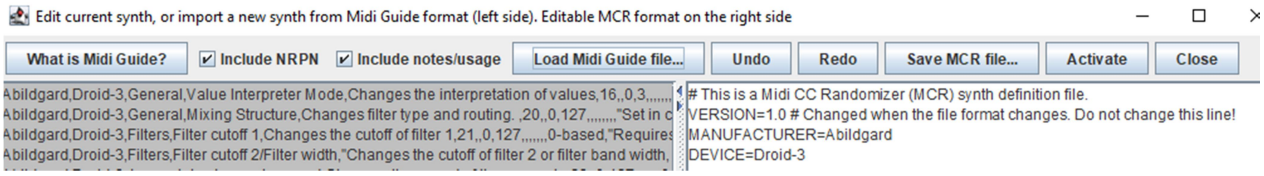
- **Comments** (anything after the first **#** on a line). Comments can also be added after the data on the next two types.
- **Tags** in the format TAG=VALUE
- **Parameters** in form of a comma-separated list of values

Creating a synth definition file from a Midi Guide file (import)

You can import a Midi Guide file, and use that to create a MCR file. Many of the Midi Guide files are missing the Section field, which means you should edit the file when importing so you can separate all the different sections correctly (having 1 section with all parameters is not very useful). That can be done by putting in the section name on each parameter line, or you can use the **SECTION** tag before each section to set the section name to use until the next **SECTION** tag is found. You might also want to disable some parameters, or adjust the min/max value before finalizing your MCR-file. Sometimes parameters belonging to the same section is spread out in the midi guide file

Import a Midi Guide file

To import a Midi Guide file, click on **Import…** at the top of the main window. This will open the import window, where you must click "**Load Midi Guide file…**". After selecting the file to import, it is loaded into the left part of the window. Use the two checkboxes "**Include NRPN**" and "**Include notes/usage**" to include or ignore NRPN-lines from the import, and to include or ignore notes / usage fields (which can be long…). The checkboxes are only used when doing a **load**, so changing them will not have any effect until you do a new **Load Midi Guide file…**.



The right part contains the MCR-format converted from the loaded file. If there were any errors loading the file, the import is aborted, and an error message should show the first line with an error. Load the file into a text editor like notepad, and investigate / correct the issue.

You can edit a converted file in the right text area (f.ex add tags needed for showing it correctly in the GUI). All relevant tags are included as commented lines. See the following chapter "**Editing a MCR file** for more details on the editing part of the import!

Editing a MCR file

Editing a MCR file can be done by 1) importing a Midi Guide file, or 2) clicking **edit** in the main window to edit the currently loaded synth.
From here the process is identical.

Note: You can close the import window, and reopen it at any time without losing data, but if you click on **Edit** in the main window or import another Midi Guide file the current content of the import window is replaced.

Undo/**Redo**: Buttons or **ctrl+Z** / **Ctrl+Y** (or **cmnd**-button on mac) when focus is in the right text field this will undo/redo any action done.
Save MCR file: will present a save dialog, that opens in the folder you last did an import from (or a save from this button)
Activate: To quickly see the effect of any change, **activate** will push the new data to the main window (if there is no errors). If it fails, it will try to explain why (including the line number with the problem), so you can correct the data and trying again. A line which is not recognized as a valid tag, or is an empty or commented (#) line is expected to be a parameter line. If the number of ",-separated" fields is not **6, 8 or 13** the line will abort the activation with an error message pointing to the line (there might be more errors further down).

When you're happy with your synth definition file, you can save it from the main window by clicking **Save…** (the folder defaults to where the .jar is located, or where you last did a load/save from the main window) or from the import window by clicking **Save MCR file…** (the folder defaults to where you last did a load or save from the import window)

The proposed file name will either be based on the imported file name, or be named ***Manufacturer-synth*_mcr.txt**.

This can be loaded from the main window when needed, or set to load when starting the app with **java -jar MidiCcRandomizer.jar your_synth_mcr.txt**

You can edit a loaded MCR file at any time, by clicking the Edit button in the main window. The import window will open and allow the current synth to be edited in the right text area.

Description of valid tags

Here is a complete description of the tags you can use to modify the GUI, or to get things to work with your synth. For fixing issues with NRPN a midi monitor (like MIDI-OX) can be useful, but a DAW can probably do the same by capturing data from your synth so you can inspect how the synth would expect it back.

Tag: **VERSION**
Not to be change by the user!

Tags: **MANUFACTURER** and **DEVICE**
Optional tags to show the manufacturer and device name in the application window.
Can also be used as suggested file name when saving.
MANUFACTURER=Arturia
DEVICE=Matrixbrute

Tag: WINDOW_*
Optional tags to set the window width and height that fits the current synth:
WINDOW_WIDTH=1000 # width in pixels
WINDOW_HEIGHT=800 # height in pixels

Tags: **WIDTH_***
The optional WIDTH_* tags sets the pixel width of following elements in the UI:
WIDTH_SECTION=120 # Section name and section buttons (default 120)
WIDTH_NAME=200 # Parameter name (default 200)
WIDTH_VALUE=100 # Value (default 100)
WIDTH_CC=80 # CC-field (default 80)
WIDTH_RANGE=120 # Range min/max (default 120)
These tags can be used if the defaults are not suitable, f.ex if the section name or parameter names are all very short, or some are very long.

Tag: **SECTIONS_ACROSS**
The optional tag **SECTIONS_ACROSS** sets the number of section buttons on each line:
SECTIONS_ACROSS=4 # Number of sections across the window

Tag: INCLUDE_SECTION
Optional tag to set the include checkbox of the current, and all following sections.
Default is YES.

INCLUDE_SECTION=YES # Set sections to included=1
INCLUDE_SECTION=NO # Set sections to included=0

Tag: INCLUDE
Optional tag to override the enable-field on all the parameters following this tagValue.
INCLUDE=YES # Set included=1 instead of the value in the enable field
INCLUDE=NO # Set included=0 instead of the value in the enable field
INCLUDE=END # Use the enable fields from here on. (Any value except YES and NO will work, not just END)

Tag: SECTION
Optional tag to set the name of the current section.
This is also a quick way to create sections in imported files that are missing a name in the section field (and therefore create one section for all parameters)
SECTION=New name # Creates a new section named "New name" with the parameters following
SECTION # Will use the name found on the parameter line.

Tag: DELAY
Optional tag to have a delay between parameters if the synth can't handle the speed from MCR.
A delay in milliseconds can be inserted after each X parameter, or if interval is not given the pause will be between each parameter.
DELAY=delay (ms), interval
DELAY=20,10 # This will create a delay of 20 ms after every 10 parameters sent
DELAY=20 # This will create a delay of 20 ms between each parameter

Tag: NRPN_ORDER
Optional tag that only applies to NRPN.
It is used to control the order of the four individual CC messages sent for a NRPN message.
Note: It is included in case it is needed in some rare cases - it has not been needed so far...

NRPN ("Non-Registered Parameter Number") is a way manufacturers can expand the number of different CC-messages and values from 128 to up to 16384.
This is done by sending 4 ordinary CC-messages: CC-99/CC-98 is parameter MSB/LSB, and CC-6/CC-38 is value MSB/LSB. Optionally CC-6 can be dropped if the value is 0 (some synths don't like to get CC 6:0)
Use this tag to swap the order MSB and LSB are sent.

Default is to send the four CC-messages in this order, which is the same most synths use:
NRPN MSB: CC-99 = 0
NRPN LSB: CC-98 = 9
Value MSB: CC-6 = 3
Value LSB: CC-38 = 116

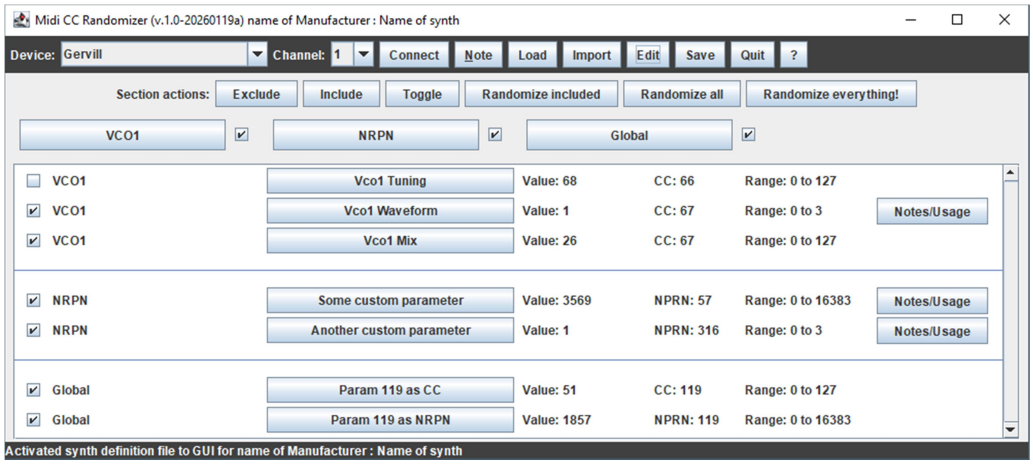
To change the order, the value are 4 fields indicating the order:
NRPN_ORDER=PM,PL,VM,VL # Same as default order (=no effect)
NRPN_ORDER=PL,PM,VL,VM # Sending LSB first
PM / PL = Parameter MSB / Parameter LSB
VM / VL = Value MSB / Value LSB
To be valid it must be 4 fields, and all 4 values must be present once.

Tag: NRPN_ONLY_LSB
Optional tag, to just send CC for NRPN value LSB, when MSB is zero.
Some synths might not behave correctly when receiving the MSB, if the parameter has a range of 0-127.
NRPN_ONLY_LSB

Tag: NRPN_TERMINATE
Optional tag, to send a "null rpn" message to terminate the NRPN parameter change.
This is mentioned as "best practice", but might have different effect on different synths.
The tag have no arguments, and can be included once anywhere in the file to enable sending the "null rpn".
NRPN_TERMINATE

The Template_mcr.txt file

```
# This is a Midi CC Randomizer (MCR) synth definition file.
VERSION=1.0 # Changed when the file format changes. Do not change this line!
MANUFACTURER=name of Manufacturer
DEVICE=Name of synth

# The following tags can customize the GUI to better fit a given synth.
# GUI layout - width in pixels for some of the elements, the 2 first are the one to change:
# WINDOW_WIDTH=1000 # initial width & height of the whole window
# WINDOW_HEIGHT=600
# WIDTH_SECTION=200 # the section button
# WIDTH_NAME=240 # the parameter name button
# WIDTH_VALUE=120 # the value column
# WIDTH_CC=80 # the CC column
# WIDTH_RANGE=120 # the range column
# SECTIONS_ACROSS=5 # Set number of section buttons on each line in the sections part.

# The following NRPN-settings might be needed for certain synths.
# NRPN_TERMINATE # Use if the synth requires CC101:127 + CC100:127 to terminate a NRPN parameter.
# NRPN_ONLY_LSB # Use if the synth only want the LSB of the NRPN parameter value (skips CC.
# Order of NRPN sent as CC: PM=CC MSB, PL=CC LSB, VM=Value MSB, VL=Value LSB
# NRPN_ORDER=PM,PL,VM,VL # Set the order of CC messages for a NRPN

# A delay can be added if there is issues changing many parameters at the same time:
# DELAY=5 # 5ms delay after each parameter
# DELAY=5,10 # 5ms delay after every 10 parameters

# The following tags can be used between the parameter lines to control how they are used:
# INCLUDE=YES # All parameters following will be included (or not, when NO)
# INCLUDE=NO  # This will override the value in the field for 'included'
# INCLUDE_SECTION=YES # Will check the Include checkbox for the current section
# INCLUDE_SECTION=NO # Will uncheck the Include box. Both applies until the next INCLUDE_SECTION tag.

# To create custom sections, use this tag:
# SECTION=Name # overrides the section name until the next SECTION tag.
# SECTION= # Empty name will cancel override, and use the section name from the line.

# The parameters. A new section is created when the section name is different from the previous line.
# There are 3 possible formats, and they can be mixed in one file:
# Include(0/1),Section,Parameter,CC,Minimum,Maximum
# Include(0/1),Section,Parameter,CC,Minimum,Maximum,notes,usage
# Include(0/1),Section,Parameter,CC,Min,Max,Use NRPN (0/1),NRPN MSB,NRPN LSB,NRPN Min,NRPN Max,notes,usage
# SECTION= # stop override section name

# Section: Vco1
0,VCO1,Vco1 Tuning,66,0,127,, # Disabled (0 in first field), don't want to randomize the tuning!
1,VCO1,Vco1 Waveform,67,0,3,Waveform shape,
1,VCO1,Vco1 Mix,67,0,127,,

# Section: NRPN-params - note quotes around note field, as it contains a comma.
SECTION=NRPN # Name of section applies to all parameters until the tag "SECTION" without any name
1,,Some custom parameter,,,,1,0,57,0,16383,"NRPN 57, range 0-16383",
1,,Another custom parameter,,,,1,2,60,0,3,"NRPN 316, range 0-2",0=left;1=middle;2=right
SECTION

# Section: Global
1,Global,Param 119 as CC,119,0,127,0,0,119,0,16383,,
1,Global,Param 119 as NRPN,119,0,127,1,0,119,0,16383,,
# End of file
```



Randomizing VST's in a DAW (or a stand-alone VST)

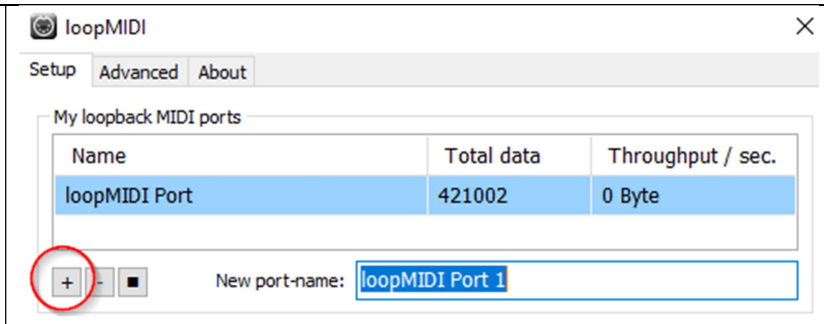
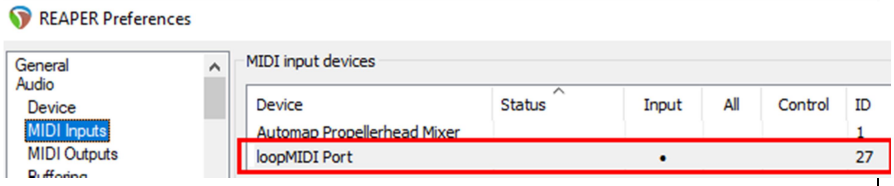
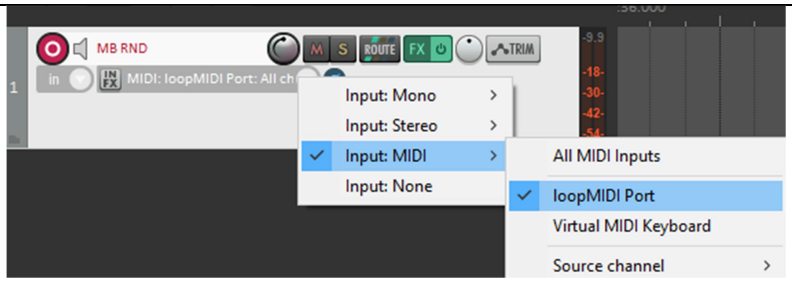
This example sets up MCR to randomize the **Arturia Mini V (Minimoog)** VST in **Reaper** (DAW), using the free **loopMIDI** to connect MCR to the VST in Reaper. This should work with any DAW. To set it up correctly, you might need to check your documentation, or use google.

To be able to let MCR connect to a DAW or other Midi software running on your pc, a **virtual loopback MIDI port** must be set up. On windows the free **loopMIDI** can be used. To set it up, follow the guide for the software you choose. For other platforms, please google "virtual loopback midi" for suitable software.

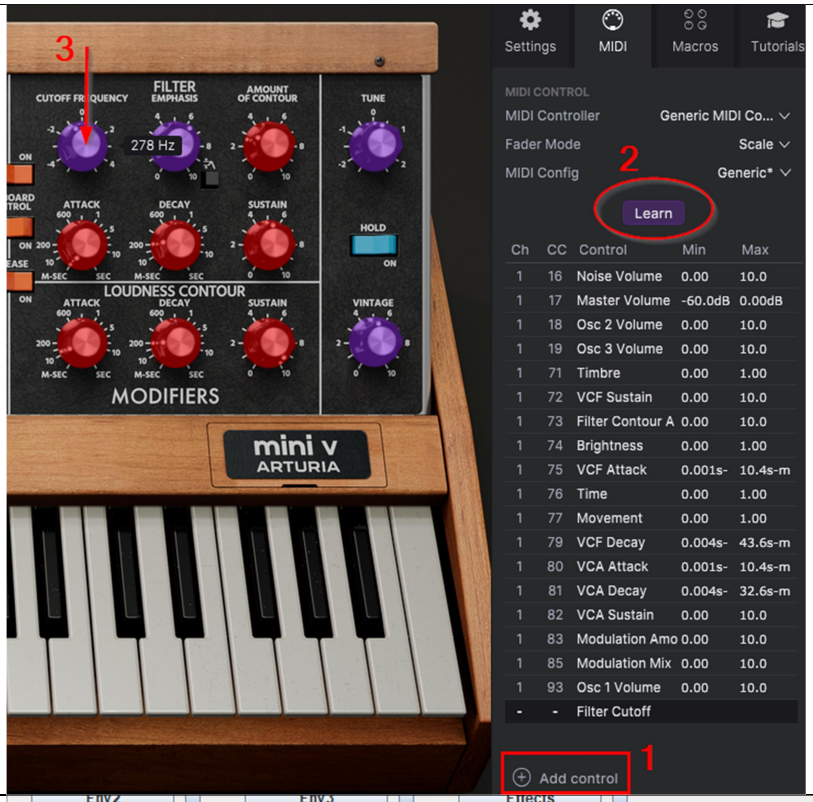
For mac, you can try this (not tested): https://support.apple.com/guide/audio-midi-setup/transfer-midi-information-between-apps-ams1013/mac

Links: https://www.reaper.fm/ and https://www.tobias-erichsen.de/software/loopmidi.html

Set up loopMIDI in Reaper on Windows

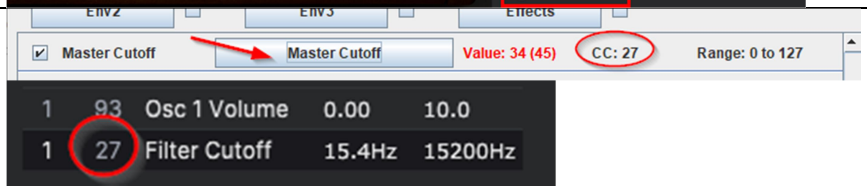
1. Add a new port under Setup. For other software, see the documentation	
2. Add the port to Reaper, in Reaper preferences, section Audio / MIDI Inputs	
3. Choose the same port in the DAW (here: Reaper). For **Source channel**, select the same channel as in MCR (or just select **All Channels**).	

4. Open MIDI-settings in the VST, then "**Add control**" (here Filter Cutoff is selected in the Add control page that opens). Click **Learn**, then click the button/knob you want to associate with the CC.



5. In MCR, select the parameter you want to use for this VST parameter. Here CC 27 is assigned to Filter Cutoff on Mini V.

   Each click on "Master Cutoff" in MB-RND will send a random value to Cutoff Freq on the Mini.



Troubleshooting

In case the program does not behave as expected, please do the following:

- See the log file located in the folder beside the jar-file and look for clues on what could be wrong.
    - It is called **log_mccrnd.txt** and contains the log for the last time you ran MCR
    - When starting MCR, this log is copied to **log_mccrnd_previous.txt**
        - This means logs exist for the **previous**, and **current** run of MCR, older logs is gone!
- Copy the log file so you have a backup that will not be overwritten.
- Try to reproduce the issue, and note down each step.
- Send this to matrixbrutepv@hotmail.com and be sure to include the corresponding log file, and the file you tried to import / load.

Log file

There is a log file available, unless turned off when running MCR with the option -NOLOGTODISC. It is located in the same folder as the MidiCcRandomizer.jar.

When starting MCR, the log file is copied to **log_mccrnd_previous.txt**, and a new log file **log_mccrnd.txt** is created. When you close MCR, **log_mccrnd.txt** will contain the log from last running MCR, and **log_mccrnd_previous.txt** contains the log from the run before that.

Be sure to make an extra copy of the logs if you experience problems. This "rolling log" is done to avoid filling the disc over time with log-data, but still be able to investigate any issues.

You can also configure MCR to write the same data as written to log to the console while the app is running. This can be switched on independent of the logging to disc, with the switch –LOGTOCONSOLE, and will work if you start MCR from the command line.

Known issues

Novation Mininova: if sending NRPN value with MSB+LSB when MSB is 0, it works the first time, the next time no change is done. To fix this, send only LSB when value is in range 0-127. Use the tag **NRPN_ONLY_LSB**. The included definition file **Novation-MiniNova_mcr.txt** are configured so it works. The same issue *might* also apply to other Novation synths.

Windows: When using multiple monitors, the windows will open on the monitor that is defined as "my main display".

Running mulltiple instances: This might mess up logging to the log file, as all running will log to the file **log_mccrnd.txt** located in the same folder as the jar.