midiBrain manual vers. 1.0

What does it do?

midiBrain records and loops arbitrary midi data synced to ReWire or to internal clock and midi clock master. One multifunction-button can control the essential looper function for easy use in performances. Incoming midi (optional) can be set to a chosen channel in order to address the events dynamically to different targets on the fly. The program keeps track of programs for all 16 midi channels to easily switch between programs of synthesizers, whether they are virtual or not.

The program is meant to sit in front of the sequencer software/synthesizer so that Fi. In case of Ableton pre-mapped midi controllers to parameters like the frequency of an *autofilter* can be looped, as well as aux-sends like all midi assignable parameters and of course notes for playing an instrument etc. . midiBrain gets events from its 3 midi sources to record/loop and alter them and the events are forwarded to a virtual midi-port (in case of using it with a sequencer software) or sent out straight to physical midi-port or any program or device which is capable of receiving and transmitting midi events. You can use this software standalone as it can be used with its internal clock and sync midi hardware to it.

A. Prerequisites

you need the current Java Runtime Environment

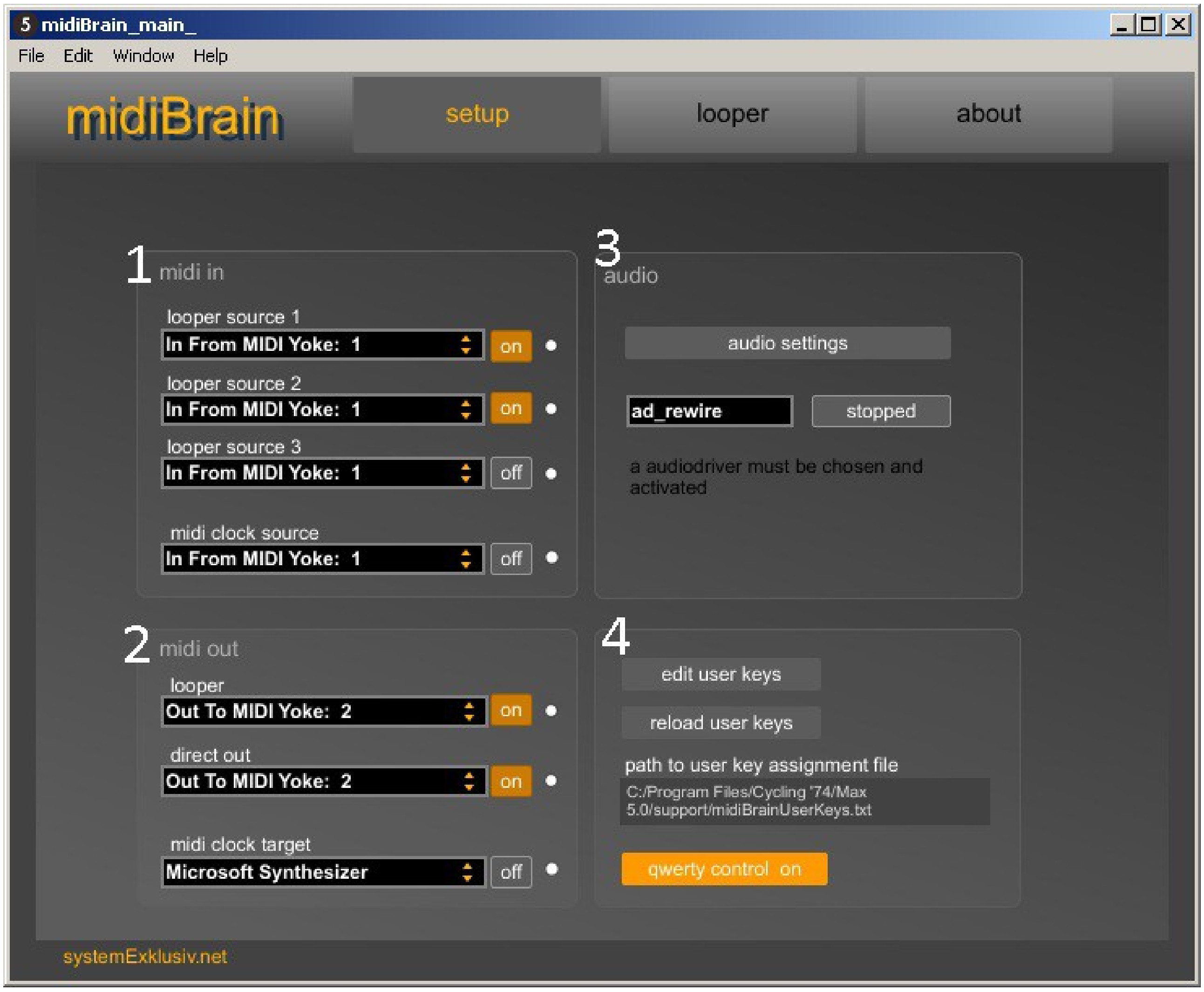
- Get Java here (its preinstalled in most cases): http://www.oracle.com/technetwork/java/javase/downloads/index.htm

- If you plan to use a software along with midiBrain you need a **virtual midi driver**, like midi yoke on PC or the preinstalled driver of the mac OS IAC with at least 1 active midi out port activated.
  Get it here (if you are on a PC): http://www.midiox.com/index.htm?http://www.midiox.com/myoke.htm

- If you are using the Max-Collective version (not the .exe version) , you need to have the Max-Runtime installed. Get it here :http://cycling74.com/downloads/
  This is for mac. In that case you need to copy the midiBrain jar to the
  Max search path of the runtime (for java/classes).

B. Setup

Before recording and looping some setting up has to be done. The following shows the *setup*-page:


5 midiBrain_main_

File Edit Window Help

midiBrain | setup | looper | about

1 midi in

looper source 1: In From MIDI Yoke: 1 — on

looper source 2: In From MIDI Yoke: 1 — on

looper source 3: In From MIDI Yoke: 1 — off

midi clock source: In From MIDI Yoke: 1 — off

2 midi out

looper: Out To MIDI Yoke: 2 — on

direct out: Out To MIDI Yoke: 2 — on

midi clock target: Microsoft Synthesizer — off

3 audio

audio settings

ad_rewire — stopped

a audiodriver must be chosen and activated

4

edit user keys

reload user keys

path to user key assignment file: C:/Program Files/Cycling '74/Max 5.0/support/midiBrainUserKeys.txt

qwerty control on

systemExklusiv.net

(Pic1: example midi settings in *midiBrain*)

Almost all controls have tool tips - you can see them when you hover the mouse over a control element. Here's the description of the settings:

1. midi in

looper source 1, 2 and 3
Here you can select arbitrary midi port for recording and looping.

midi clock source
If you plan to sync midiBrain's transport to an external midi clock, than you should select the midi in port which receives the clock (like from a drum computer).

2. midi out

looper
If some midi events have been recorded in the looper unit, the played back events come out this port.

direct out
All data that is coming in from the sources is directly forwarded to the direct out (except they are part of predefined user keys)

3. audio

An audio driver must be chosen in order to make the looper and metronome work properly. If you are using midiBrain alone (without e.g. a sequenzer-SW) any active driver is valid. In case of using it together with Ableton (or any other ReWire-host) choose „ReWire“ as driver and turn it on.

4. Controlling midiBrain

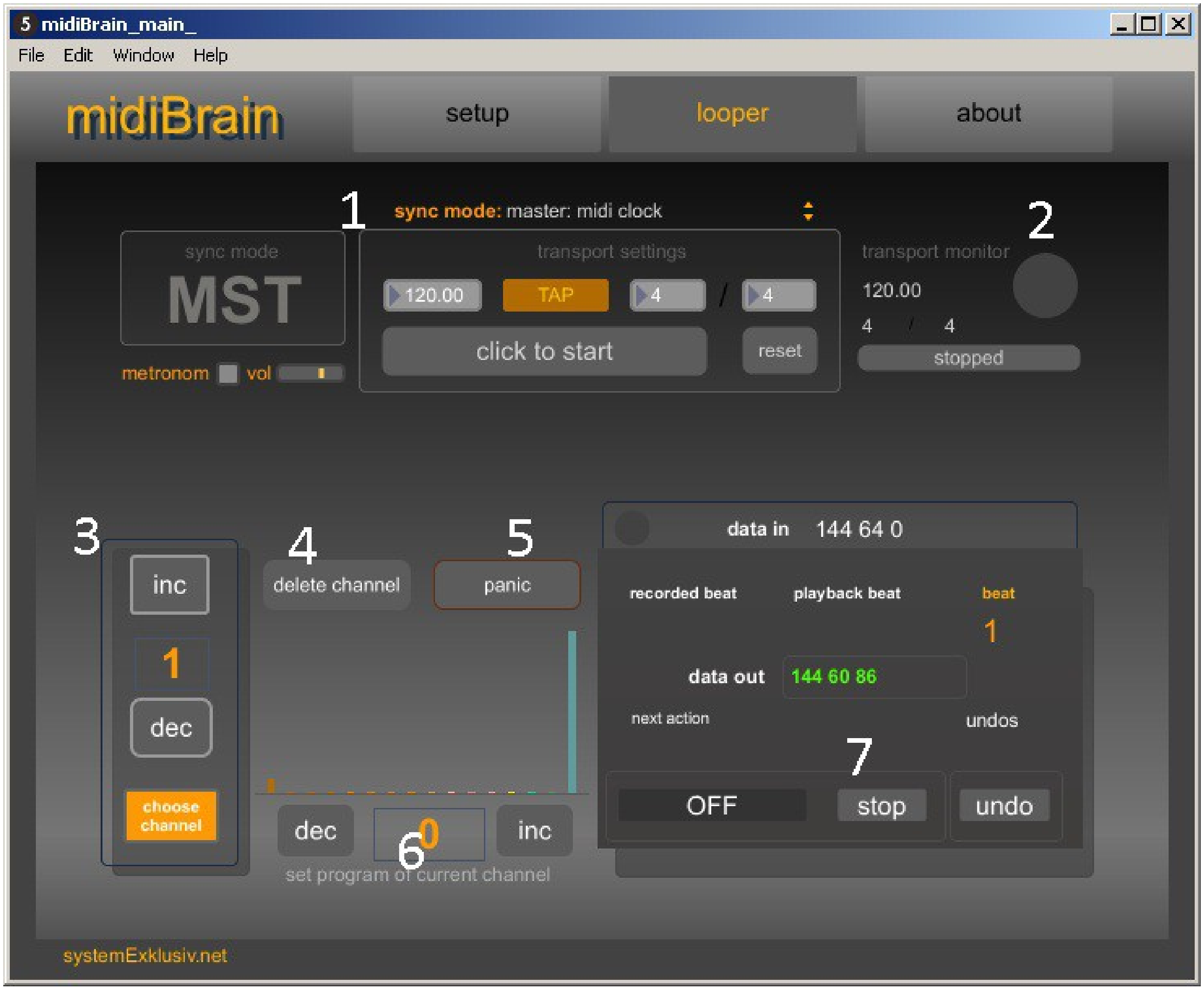
Midi

Midi that is coming from the looper-sources is filtered first and checked if they are part of predefined user keys. The definition of the events for controlling the looper and more happen in a text file which is editable in midiBrain (or by any text editor). Click on 'edit user keys' and a text editor opens where assignments can be made (see '**D Editing The User Key Assignments**).

Qwerty-Control

You can use a common compuer keyboard to control the program. A tool tip shows the hard coded keys/function (e.g. Space for REC/PLAY/OVR).

C. The Looper



(Pic 2: looper page overview)

The following described functions can almost all be mapped to midi events or controlled by a qwerty-keyboard.

1. Sync-Mode
You can run the program as slave or master. Depending on the chosen mode the transport controls differ. If you choose slave:rewire, you have no controls, all is set by the host software. If you choose master:midiclock (like in the pic) you have full transport controlls.

2. Transport Monitor
This sections shows midiBrain's transport state. Its just for monitoring. Available transport-settings like tempo or signature can be set left hand under *transport settings*. The available settings depending on the selection of the drop-down box *sync mode* .

3. Choose Channel Segment
If active, the incoming midichannel of all events (except CC by default– but this can be configured) is ignored and the chosen one – by typing in a number or pressing *inc* / *dec* – is taken. Formatted events like this are sent to the looper and to direct out.

4. Delete Channel
Depending on the channel selected under 3. (*choose channel*) all midi data is erased in the looper recorded on this channel.

5. Panic
Sends note off out on all midi channels. This can be useful if a note off hangs, no note off is recorded. I tried to make it robust, so that it is checked all the time if there are missing note offs so on, but just in case..

6. Program Change By Chosen Channel
For the channel chosen under 3. a program change can be sent. The *inc* and *dec* buttons lead to increase/decrease the program and sent the newly set program out. You can use the number box as well. Over the buttons is a visualization off the program counter for each channel.

7. The Looper Unit
The looper can record all midi events that are used for playing and performing (note on/off, control change, pitchbend etc.). It sits after the *choose channel* segment and records altered channels (if the *choose channel* segment is turned on). The *program changes* which can optionally sent for the chosen channel (like in 6.) are not recorded.

7.1 *REC/PLAY/OVR*-Button
The main controlling button is the multifunctional button: *RECPLAYOVR*. The transitions of the button states are described in the following:

OFF >> REC >> PLAY >> (OVR >>PLAY)*
Its initial state is *OFF*. If pressed next time, it records the first loop (state: *REC*). The recording starts at the first on-beat of a measure. The measure can be a 4/4 or 7/8 (or any other signatures), all transition between the states happen on the first on-beat. Pressing it next time it changes to *PLAY*. The first recording determines the total length of the loop. In *PLAY*-mode no events are being recorded, You can play a solo on top of your loop or look for a good sound or simply avoid to record something. Pressing it next time changes its state to *OVR*-mode. Now events are recorded on top of the prior recording. From this point on it cycles between *OVR-PLAY* all the time until the whole loop has been erased.

7.2 *STOP*-Button

As you might guessed: it stops the playback of the looper. Pressing the *REC/PLAY/OVR*-Button again, starts its Playback on the next on-beat. The state of this button is set to *PLAY*.

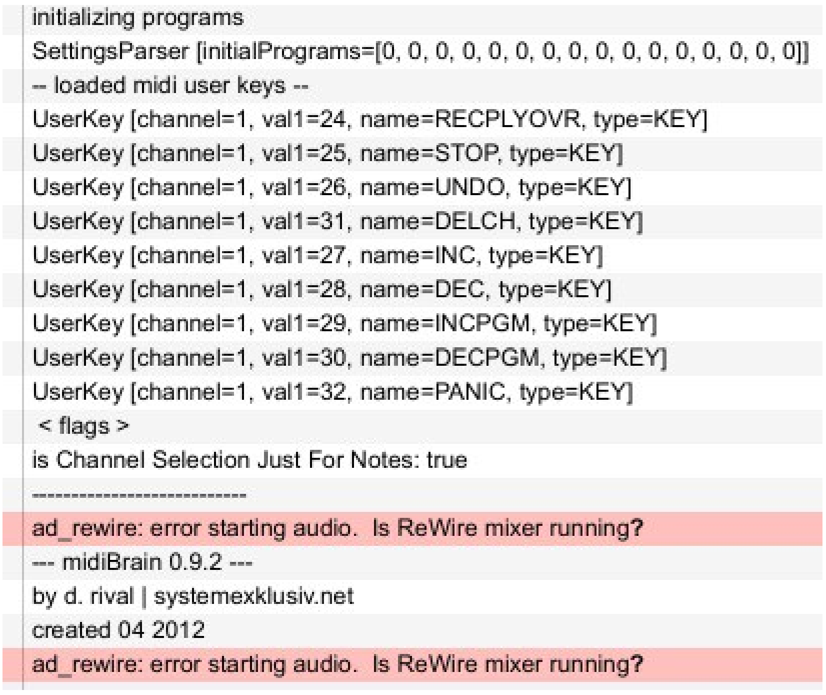
7.3 *UNDO*

This button has 2 functions. If The looper returns from *OVR*-mode back to *PLAY*-mode the recordings taken can be erased by *UNDO*. In the GUI you can watch below the label *undos* the number 2. There is just 1 *UNDO*-Step possible and if the looper returns the next time from *OVR*-mode, the prior UNDO is overridden by the new one. If pressed once you can see written below the *undos*-label *?clear?* - which means pressing it again deletes the whole content of the looper and sets it back to *OFF*-mode. Another way of deleting events from the looper is using the *Delete Channel* – function like described under 4.

D Editing The User Key Assignments

D.0 Controlling midiBrain With Midi

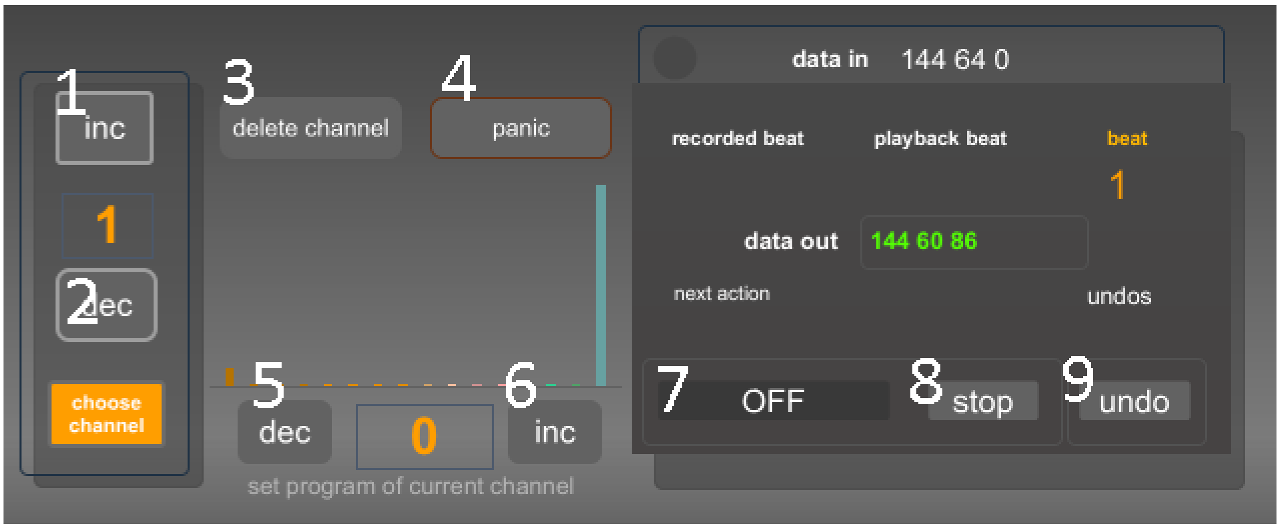
A text file is used to assign midi events to program functions. CC and note on numbers can be used in the assignments. When the program starts this file is parsed and the program is set up accordingly. The assigned events are not forwarded or recorded in the looper but they are filtered out and trigger the midiBrain's functions like *UNDO*. After program start you can see a summery of the settings in the Max-window. Hit **ctrl + m** to open the MAX window:



```
initializing programs
SettingsParser [initialPrograms=[0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]]
-- loaded midi user keys --
UserKey [channel=1, val1=24, name=RECPLYOVR, type=KEY]
UserKey [channel=1, val1=25, name=STOP, type=KEY]
UserKey [channel=1, val1=26, name=UNDO, type=KEY]
UserKey [channel=1, val1=31, name=DELCH, type=KEY]
UserKey [channel=1, val1=27, name=INC, type=KEY]
UserKey [channel=1, val1=28, name=DEC, type=KEY]
UserKey [channel=1, val1=29, name=INCPGM, type=KEY]
UserKey [channel=1, val1=30, name=DECPGM, type=KEY]
UserKey [channel=1, val1=32, name=PANIC, type=KEY]
 < flags >
is Channel Selection Just For Notes: true
--------------------------
ad_rewire: error starting audio.  Is ReWire mixer running?
--- midiBrain 0.9.2 ---
by d. rival | systemexklusiv.net
created 04 2012
ad_rewire: error starting audio.  Is ReWire mixer running?
```

The first line after ' initializing programs' shows the default programs for each channel.
Each line starting with UserKey […] is a single midi assignment. You can edit the assignments while running the program and reload them. Its a very simply syntax.

In the following I will describe the Keywords and the values of the file. At the beginning you will see a screen shot of the *looper*-page with white numbers. Each of the indexed buttons can be controlled by midi.

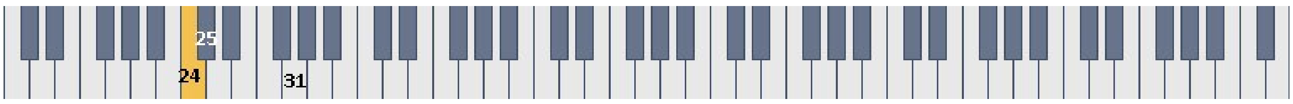


(Pic 3: indices of midi assignable buttons)

For instance starting at (7) the multi button, which has the keyword 'RECPLYOVR'. The example assignment is below. The button is triggered by the event 'key' on channel 1 with the pitch number '24' which is a C0..

1. # the multi button of the looper
2. RECPLYOVR key 24 channel 1

Line 1 starts wit a '#'. Lines introduced with this are comments and are ignored by the parser. Line 2 is the actual assignment. In the default assignments, all expected events are *key.*



(Pic 5: illustration of pitch numbers and actual keys)

You can see the associated pitch number with the key on a midi keyboard. C0 equals 24 which is displayed above. In the default file (which you can download separately) all controls are defined between number 24 to 32. The idea was to use an octave one is not likely to play (– you could use very high pitch numbers as well or of course smthg else than a midi keyboard). In the following the rest of the default assignments:

Assignments	**Reference in Pic 3**
STOP key 25 channel 1	(8)
UNDO key 26 channel 1	(9)
DELCH key 31 channel 1	(3)
INC key 27 channel 1	(1)
DEC key 28 channel 1	(2)
INCPGM key 29 channel 1	(5)
DECPGM key 30 channel 1	(6)
PANIC key 32 channel 1	(4)

(Table 1: default user key assignments)

Each line is one assignment or a comment. An assignment starts with a keyword like the above, an event-type, an event number, the word *channel* and a digit (1-16).

Lets assign a *control change* event number 60 on channel 3 for the multifunction-button:

1. RECPLYOVR cc 60 channel 3

Now the button is triggered by a new event. The value is omitted and is expected to be 127 (this is the case for all *cc* and *note* assignments). the *key* – event is special as its like a note but if the value is greater than zero its considered to be a hit. The *UserKeys*-file is commented a lot.

Possible event types are *cc*, *note* and *key*. *Key* is good if you use a midi keyboard for controlling the program, because every note on of the specified pitch number is triggering the function. It would be difficult to hit a specified velocity exactly. Control change (cc) or note are expecting 127 as value for triggering the function. This is useful if you have a midi controller and you can configure the value (the controller sends along with the status and number (e.g. bcr2000..)).

D.2 More settings

At the bottom of the file there 2 more setting options:
The first one:

IS_CHANNEL_SELECTION_JUST_NOTES true

If set to *true* (default), the midi channel of all but CC-events are altered by the *choose channel* segment (see Pic 3, 1 and 2) – if you want your connected controllers not to change the channel (so that they stay on their original channel). If set to *false* the channel of CC will be changed as well and controllers sending CC-events will send through *midiBrain* on a different channel. This can be useful to use Fi. Just one encoder to control up to 16 different Instrument parameters.

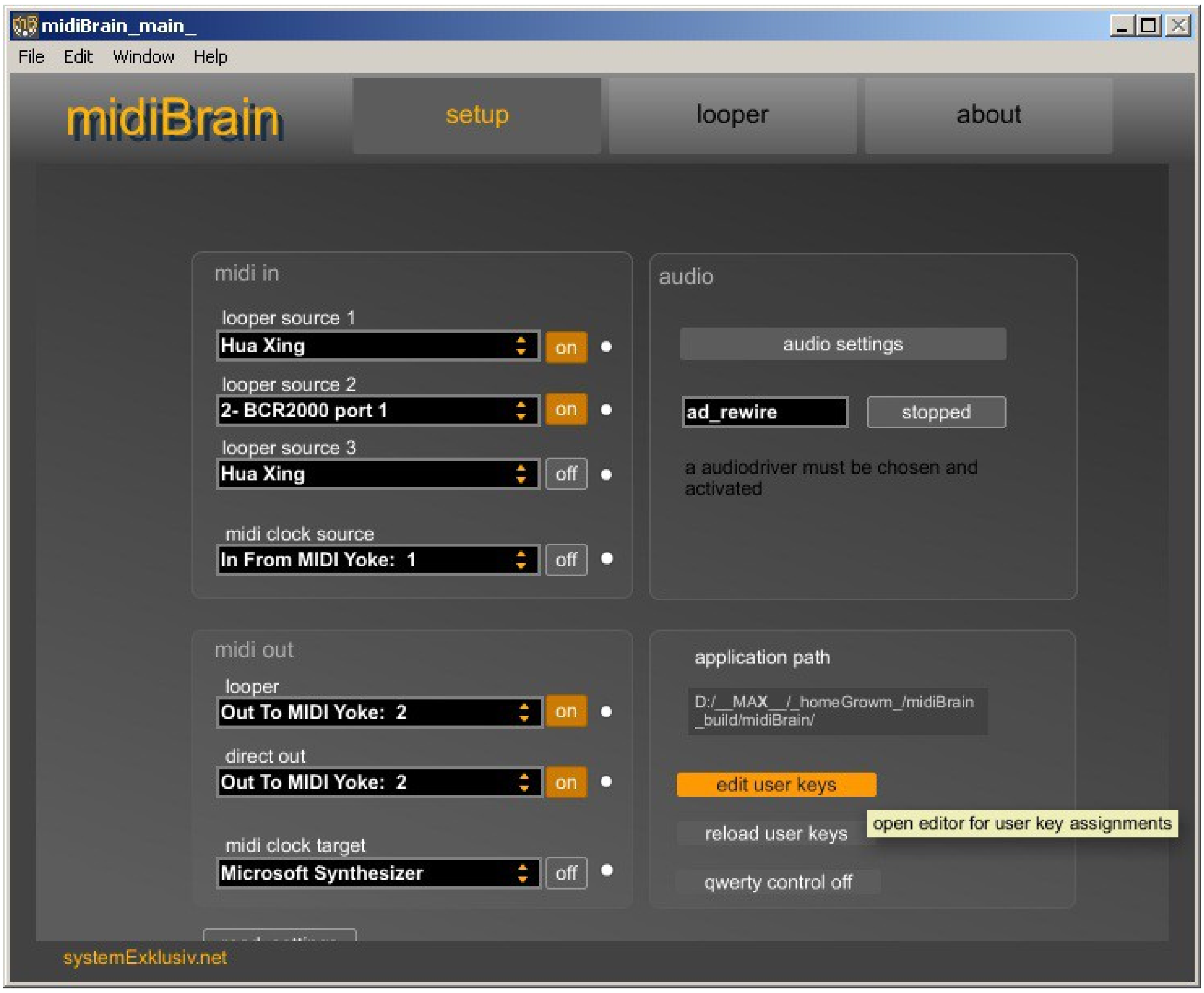
And the second one:

After the keyword *INITIAL_PROGRAMS* 16 digits in range 0-127 follow. These numbers specify the initial programs for each channel from 1 to 16 (see Pic, above 3 and 4).

INITIAL_PROGRAMS 11 0 0 0 0 0 0 0 0 0 0 0 0 0 0 127

D.3 Editing And Reloading The Settings

To Edit the assignments change to the *setup*-page and click on the 'edit user keys'-button like displayed below:



midi in

looper source 1: Hua Xing — on

looper source 2: 2- BCR2000 port 1 — on

looper source 3: Hua Xing — off

midi clock source: In From MIDI Yoke: 1 — off

audio

ad_rewire — stopped

a audiodriver must be chosen and activated

midi out

looper: Out To MIDI Yoke: 2 — on

direct out: Out To MIDI Yoke: 2 — on

midi clock target: Microsoft Synthesizer — off

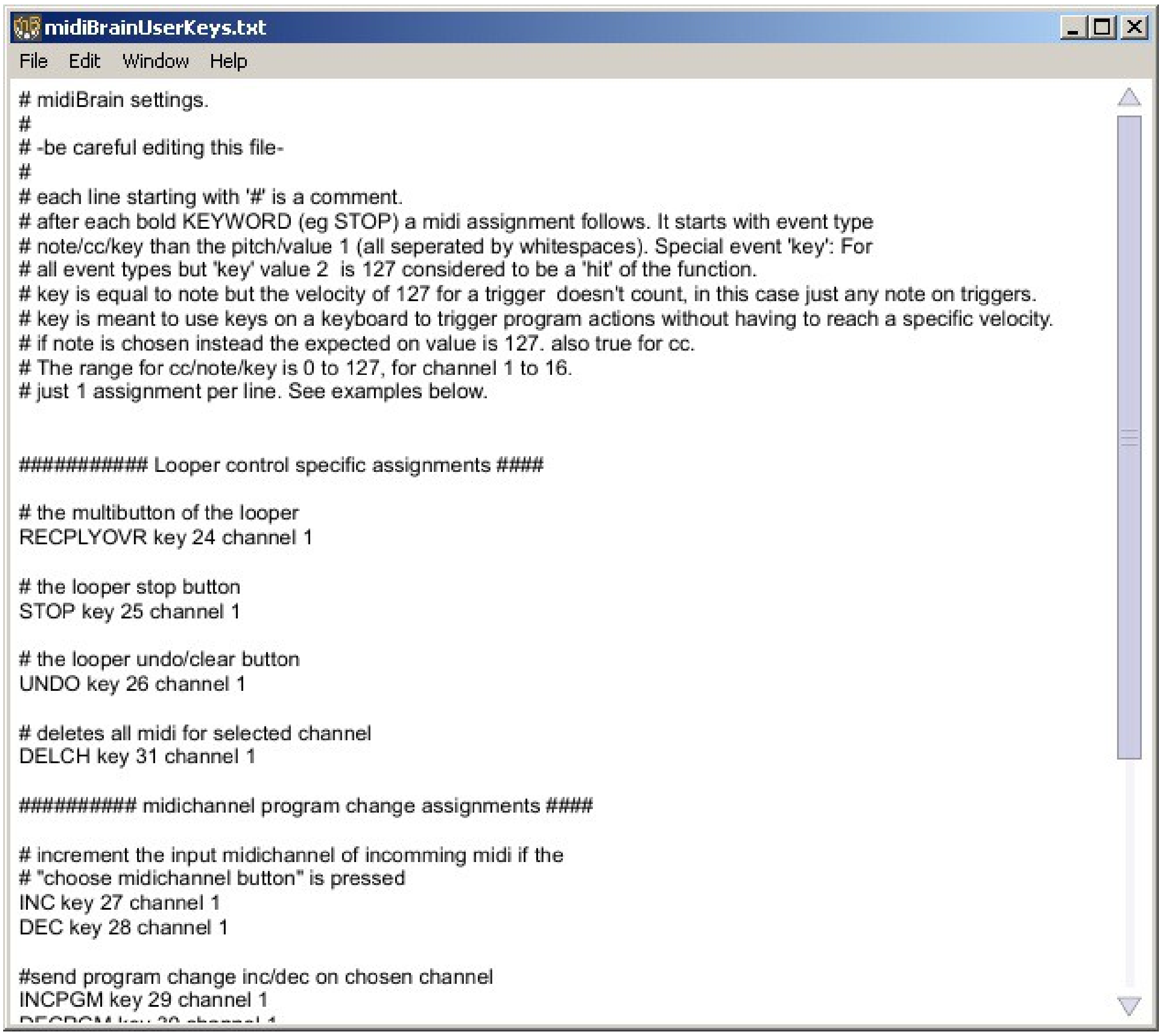
application path

D:/__MAX__/_homeGrown_/midiBrain_build/midiBrain/

open editor for user key assignments

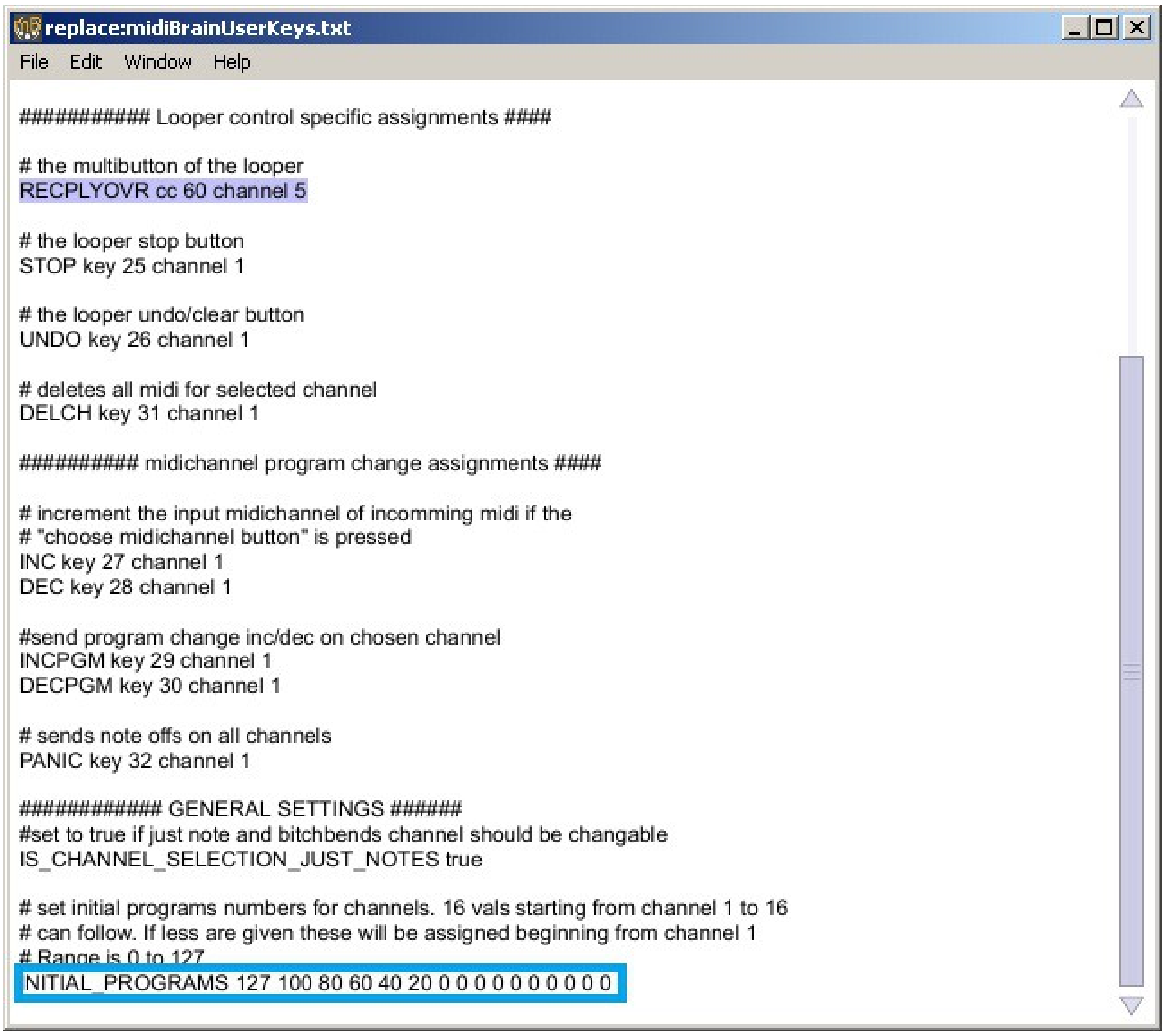
(Pic 4: edit user keys button)

A text editor opens with the content of the File UserKeysSettings.txt:



```
# midiBrain settings.
#
# -be careful editing this file-
#
# each line starting with '#' is a comment.
# after each bold KEYWORD (eg STOP) a midi assignment follows. It starts with event type
# note/cc/key than the pitch/value 1 (all seperated by whitespaces). Special event 'key': For
# all event types but 'key' value 2  is 127 considered to be a 'hit' of the function.
# key is equal to note but the velocity of 127 for a trigger  doesn't count, in this case just any note on triggers.
# key is meant to use keys on a keyboard to trigger program actions without having to reach a specific velocity.
# if note is chosen instead the expected on value is 127. also true for cc.
# The range for cc/note/key is 0 to 127, for channel 1 to 16.
# just 1 assignment per line. See examples below.


############ Looper control specific assignments ####

# the multibutton of the looper
RECPLYOVR key 24 channel 1

# the looper stop button
STOP key 25 channel 1

# the looper undo/clear button
UNDO key 26 channel 1

# deletes all midi for selected channel
DELCH key 31 channel 1

########### midichannel program change assignments ####

# increment the input midichannel of incomming midi if the
# "choose midichannel button" is pressed
INC key 27 channel 1
DEC key 28 channel 1

#send program change inc/dec on chosen channel
INCPGM key 29 channel 1
```

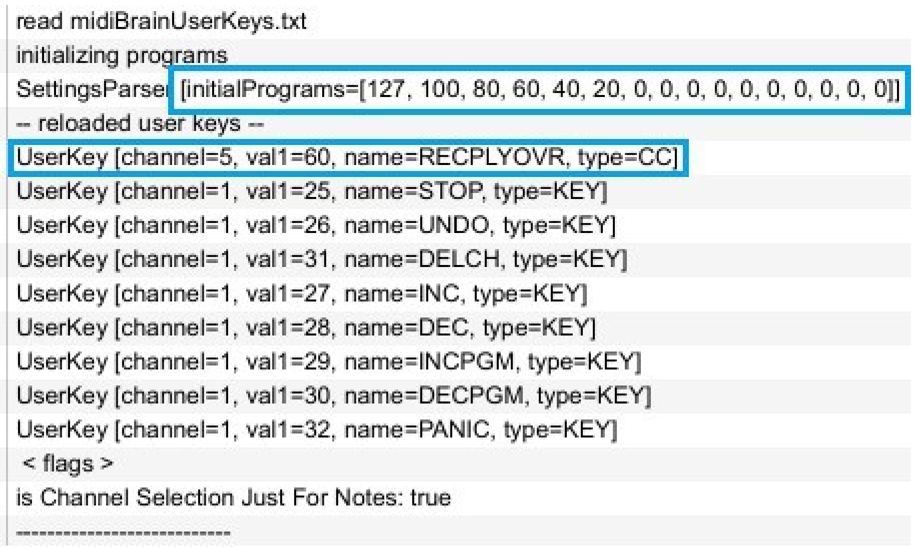
In the next screen shot I made some changes to the file. I change the event which triggers the multifunction-button of the looper (*RECPLYOVR*) and the initial programs (INITIAL_PROGRAMS). *RECPLYOVR* is now waiting for a control change number 60 on channel 5 (with value 2 of 127 – you can't change that). The initial programs are no longer 0 for all channels. For channel 1 its now 127, for ch 2 it's 100, ch 3 80 .. till channel 6 which gets the number 20. The remaining channels (7-16) stay initially on program number 0. The modified file is displayed below:



```
########### Looper control specific assignments ####

# the multibutton of the looper
RECPLYOVR cc 60 channel 5

# the looper stop button
STOP key 25 channel 1

# the looper undo/clear button
UNDO key 26 channel 1

# deletes all midi for selected channel
DELCH key 31 channel 1

########## midichannel program change assignments ####

# increment the input midichannel of incomming midi if the
# "choose midichannel button" is pressed
INC key 27 channel 1
DEC key 28 channel 1

#send program change inc/dec on chosen channel
INCPGM key 29 channel 1
DECPGM key 30 channel 1

# sends note offs on all channels
PANIC key 32 channel 1

############ GENERAL SETTINGS ######
#set to true if just note and bitchbends channel should be changable
IS_CHANNEL_SELECTION_JUST_NOTES true

# set initial programs numbers for channels. 16 vals starting from channel 1 to 16
# can follow. If less are given these will be assigned beginning from channel 1
# Range is 0 to 127
NITIAL_PROGRAMS 127 100 80 60 40 20 0 0 0 0 0 0 0 0 0 0
```

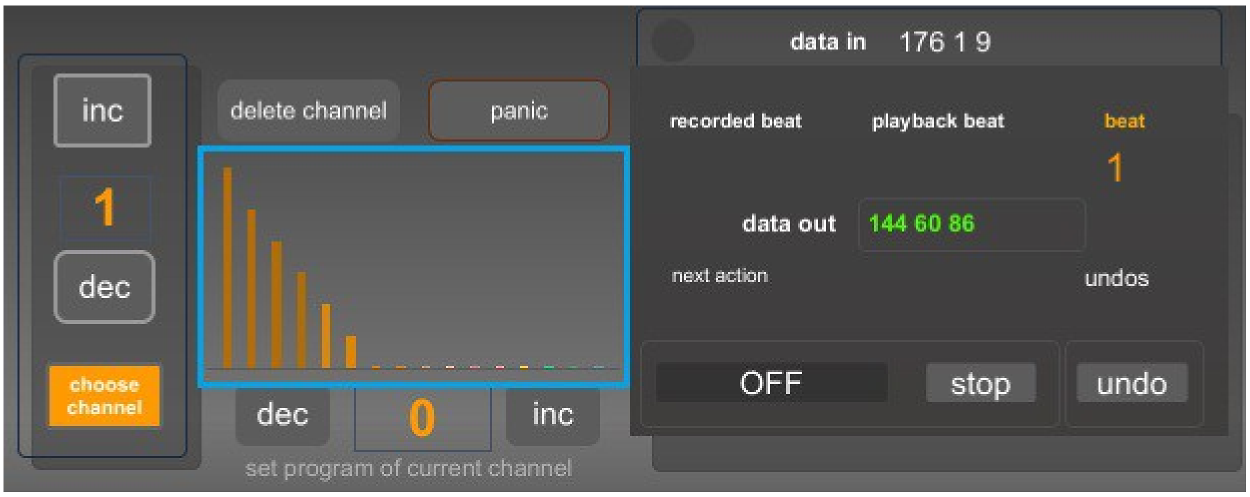
In the text editor click on 'file' > ''save' and browse to the program folder (*..\midiBrain*) and replace the file *midiBrainUserKeys.txt.* After that close the editor.

Back in the midiBrain's *setup*-page you have to reload the modified settings by clicking the 'reload user keys' – button. Now hit **ctrl + m** to open the MAX window. You can see the currently reloaded assignments:



```
read midiBrainUserKeys.txt
initializing programs
SettingsParser [initialPrograms=[127, 100, 80, 60, 40, 20, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]]
-- reloaded user keys --
UserKey [channel=5, val1=60, name=RECPLYOVR, type=CC]
UserKey [channel=1, val1=25, name=STOP, type=KEY]
UserKey [channel=1, val1=26, name=UNDO, type=KEY]
UserKey [channel=1, val1=31, name=DELCH, type=KEY]
UserKey [channel=1, val1=27, name=INC, type=KEY]
UserKey [channel=1, val1=28, name=DEC, type=KEY]
UserKey [channel=1, val1=29, name=INCPGM, type=KEY]
UserKey [channel=1, val1=30, name=DECPGM, type=KEY]
UserKey [channel=1, val1=32, name=PANIC, type=KEY]
 < flags >
is Channel Selection Just For Notes: true
---------------------------
```

If you change to midiBrain's *looper*-page you can see the initial programs in the programs-graph as well (marked with the blue box):



If one of the button-functions is triggered by midi, the roundness of the button corners changes for indication.

E Some Notes

I know midi looping is not really something new but I build something which is fun to play with in my imagination.

I hope this short manual helps to use the SW. Please contact me if you have problems to use it under david@systemexklusiv.net.

Happy looping

david