MT128

Event & Rules File Format.

This document describes the file format used in Event & Rules pages of the MT128

Table of Content:

MT128 Event & Rules:

The MT128 project include one XML file to store all Event & Rules of the project. See EventRules_Base_{XXXX_GUID_XXXX}.xml in MT128 project folder:

This document will describe file format used to store and possibly import and export table of rules and event.

General XML structure of the EventRules_Base_{XXXX_GUID_XXXX}.xml file:

```
<?xml version="1.0" encoding="utf-8"?>
<VBAudioEventAndRules>

</VBAudioEventAndRules>
```

Event & Rules can contain different kind of event & rules (like Date Event, MIDI Rules etc…). able to execute a script to perform different things.

SURVEY AND LOG
It is possible to activate a log service in startup script by the following instruction:

`SET “EventAndRulesLOG= C:\MyEventAndRulesLog.txt”`
This instruction activates and sets the file used to log events and rules being executed days after days (since it is direct file access, the related disk must be running without sleep mode).

Date Event.

The Date Event is made to generate an action on a given date range at a specific time or periodic time. All Date Event are stored in a section **VBAudioDateEventsList**.

```
<VBAudioDateEventsList>
        <VBAudioDateEvent id='0' enable='1'>
        …
        </VBAudioDateEvent>
        <VBAudioDateEvent id='1' enable='1'>
        …
        </VBAudioDateEvent>
        <VBAudioDateEvent id='2' enable='1'>
        …
        </VBAudioDateEvent>
        <VBAudioDateEvent id='3' enable='1'>
        …
        </VBAudioDateEvent>
        <VBAudioDateEvent id='4' enable='1'>
        …
        </VBAudioDateEvent>
</VBAudioDateEventsList>
```

Each Date Event is stored in an xml section, with a unique id (INT32) and an Enable status as is.

```
        <VBAudioDateEvent id='0' enable='1'>
        …
        </VBAudioDateEvent>
```

A complete Date Event item is stored in this XML format:

```
<VBAudioDateEvent id='0' enable='1'>
        <DateOriginationUTC year='2018' month='9' day='26' week='39' dayofweek='2' hour='7'
minute='50' second='14' />
        <DateEventDate1 year='0' month='0' day='0' week='0' dayofweek='0' hour='9' minute='20'
second='0' dateon='1' datestart='1'/>
        <DateEventDate2 year='0' month='0' day='0' week='0' dayofweek='0' hour='19' minute='0'
second='0' dateend='1'/>
        <DateEventTime hour='0' minute='15' second='0' modeevery='1' />
        <DateEventName>Music</DateEventName>
        <DateEventSubName>Room 4</DateEventSubName>
        <DateEventGroup>GROUP1</DateEventGroup>
        <DateEventScript>SoundPad.Button(9).PLAY=1</DateEventScript>
</VBAudioDateEvent>
```

All data not set, are set to ZERO by default (or set to empty string for string field).

Minimal Date Event.

The minimal date event is given by a time in hour, minute and second and a mode:

- AT TIME
- EVERY PERIOD OF TIME (based on 00:00:00)

```
<VBAudioDateEvent id='0' enable='1'>
	<DateEventTime hour='0' minute='15' second='0' modeevery='1' />
	<DateEventScript>SoundPad.Button(9).PLAY=1</DateEventScript>
</VBAudioDateEvent>
```

But it's recommended to define a Name at least (to sort out event and identify it more quickly). Label can be define for Name, SubName and Group.

```
	<DateEventName>Music</DateEventName>
	<DateEventSubName>Room 4</DateEventSubName>
	<DateEventGroup>GROUP1</DateEventGroup>
```

Date Range.

Event can be valid in a specific period, or after a start date, or until an end date, or on specific day only (e.g. on Monday only). These conditions are given by the two lines below defining a possible start date and a possible end date:

```
	<DateEventDate1 year='0' month='0' day='0' week='0' dayofweek='0' hour='9' minute='20'
second='0' dateon='1' datestart='1'/>
	<DateEventDate2 year='0' month='0' day='0' week='0' dayofweek='0' hour='19' minute='0'
second='0' dateend='1'/>
```

```
dateon='1' says if we use date range condition
datestart='1' says if we use date1 as start date
dateend='1' says if we use date2 as end date
```

value range to define a date:

Date field	Default / off	min	max
Year	0 = any	2000	9999
Month	0 = any	1	12
day	0 = any	1	31
week	0 = any	1	53
dayofweek	0 = any	1 (Monday)	7 (Sunday)
hour	0	0	23
minute	0	0	59
second	0	0	59

Marker List CSV file.

The EXPORT page allows exporting Marker List as CSV File (separator is TAB):

UID	Cue	Position	Name	Type
1	1	00:00:00.000		LP
11		00:01:55.134		L
10		00:01:56.025	SoundPad	L
12		00:02:00.947		LS
2	2	00:03:21.556		LP
17		00:04:10.134		I
18		00:04:45.759		O
13	3	00:05:30.759		LP
16		00:06:27.946		L

Markers are listed in timeline order (sorted by position time code) and identified by a unique identifier (UID).

CUE number gives the CUE index as it is displayed in the MT128 CUE Panel.

The Time code is in the time format **Hour : Minute : Second . millisecond**

The type gives the Marker properties:

L: Locator.
P: Cue Play.
S: Cue STOP.
B: Loop Beginning.
E: Loop End.
I: Punch In.
O: Punch Out.

Event Script

In the **DateEventScript** section any text can be used to create requests for the MT128 or for other system functions. Each instruction line must be separated by a ';' (all index are '1' based).

SOUND PAD Instruction set

Play the Sound number 6
```
SoundPad.Button(6).Play =1;
```

Stop the Sound number 6
```
SoundPad.Button(6).Play =0;
```

Stop All SoundPad Sounds
```
SoundPad.StopAll =1;
```

Load an audio file in SoundPad Button
```
SoundPad.Button(1).Load =”F:\MediaFolder\audiofile.wav”;
```

PLAYLIST Instruction set

Start first playlist
```
Playlist(1).Play =1;
```

Stop first playlist
```
Playlist(1).Play =0;
```

Stop All Playlist
```
Playlist.StopAll =1;
```

To Select the Sound item 5 in PLaylist 2
```
Playlist(2).Select =5;
```

To Play the Playlist 2 by the Sound item 7
```
Playlist(2).Sound(7).Play =1;
```

Load an audio file in Playlist Item:
```
Playlist(1).Sound(4).Load =”F:\MediaFolder\audiofile.wav”;
```

TIMELINE Instruction set

Play TimeLine
```
Timeline.Play =1;
```

Play TimeLine Cue index 2 (cue index – as displayed in the CUE Panel).
```
Timeline.Cue(2).Play =1;
```

Play TimeLine Marker ident 2 (marker ident).
```
Timeline.Marker(2).Play =1;
```

Goto TimeLine Cue index 2 (cue index).
```
Timeline.Cue(2).Goto =1;
```

Goto TimeLine Marker ident 2 (marker ident).
```
Timeline.Marker(2).Goto =1;
```

Stop TimeLine
```
Timeline.Stop =1;
```

Pause TimeLine
```
Timeline.Pause =1;
```

Rec TimeLine
```
Timeline.Rec =1;
```

TimeLine REW
```
Timeline.Rew =1;
```

TimeLine FF
```
Timeline.Ff =1;
```

Set TimeLine chase
```
Timeline.Chase =1 / 0;
```

Goto TimeLine (hour, minute, second. and optionally millisecond).
```
Timeline.Goto =00:00:00;
Timeline.Goto =00:00:00.000;
```

TRACKS Instruction set

Set the track name
```
Track[i].Label = myName;
```

Remove Track name
```
Track[i].Label = “”;
```

Use Line Check to listen to a input (in PFL BUS).
```
Track[i].LineCheck = 1 or 0;
```

Arm Track.
```
Track[i].Arm = 1 or 0;
```

Use PFL (button in matrix mixer) to listen to a input (in PFL BUS).
```
Track[i].PFL = 1 or 0;
```

MIXER Instruction set

Set Matrix parameters by Strip:
```
Matrix.Strip[j].Gain = -6.0; (j:1 based Strip index, 0 = ALL)
Matrix.Strip[j].Arm = 1 / 0;
Matrix.Strip[j].PFL = 1 / 0;
Matrix.Strip[j].Mute = 1 / 0;
```

Gain and Mute can be relatively set, example:
```
Matrix.Strip[j].Gain += 3;
Matrix.Strip[j].Gain -= 3;
Matrix.Strip[j].Mute += 1;
```

Set Mixes parameters by Mixer and by Strip:
(Mixes index: 0 = Stereo Master, 1 = AUX1, 2 = AUX2…)
```
Matrix.Strip[j].Gain = -6.0; (j:1 based Strip index, 0 = ALL)
Mixes[i].Strip[j].PFL = 1 /0;
Mixes[i].Strip[j].Solo = 1 /0;
Mixes[i].Strip[j].Mute = 1 /0;
Mixes[i].Strip[j].Pan = -100.0 – 0 - +100.0
```

Gain, Solo, Mute and Pan can be relatively set, example:
```
Matrix.Strip[j].Gain -= 3; (j:1 based Strip index, 0 = ALL)
Mixes[i].Strip[j].Solo -=1;
Mixes[i].Strip[j].Mute -=1;
Mixes[i].Strip[j].Pan +=30
```

MASTER Instruction set

Set master parameters by Bus:
(Bus index : 0 = Main, 1 = AUX1, 2 = AUX2… 9 = PFL)

```
Main[i].Gain = -6.0f
Main[i].On = 1 / 0;
Main[i].Mute = 1 / 0;
Main[i].PAN = -100.0 – 0 - +100.0
```

All field can be relatively set, example:

```
Main[i].Gain += 6.0f
Main[i].On += 1;
Main[i].Mute += 1;
Main[i].PAN += 50;
```

System Instruction set

System Command

function Name	Value Type	Remark	Ver.
`System.KeyDown(szKey)`	String		1
`System.KeyUp(szKey)`	String		1
`System.KeyPress(szKey)`	String	Send Key Down + Key Up	1
`System.Execute(exe, dir, arg)`	Strings		1

These commands are not sent to Voicemeeter but directly to operating system.

System Execute

This function works like a “CreateProcess” or SheelExecute under windows and allow to start any application with a command line argument.

```
System.Execute(szprogram, szworkdir, szcommand);
```

Example to open a web page with the internet explorer:

```
System.Execute("C:\Program Files\Internet Explorer\iexplore.exe",
                  "","-new www.voicemeeter.com");
```

Special chars like double quotes can be inserted by this sequence %’ (percent + simple quote): then %’ will be replaced by “. To insert a percent char, simply enter it double: then “%%” will be replaced by a single ‘%’.

ENVIRONMENT VARIABLES

It is also possible to use system environment variable by using **%envname%** syntax.

Example to run the Microsoft WRITE Editor application

```
System.Execute("%windir%\write.exe","%TMP%","");
```

RUNNING DOS APPLICATION

To run command line program, you need to launch cmd.exe with /K command to specify you want to execute the command after…

Example to run ipconfig in a DOS window:

```
System.Execute("%windir%\system32\cmd.exe","%windir%\system32","/K ipconfig");
```

/C Carries out the command specified by string and then terminates
/K Carries out the command specified by string but remains

Example to ping your internet router (usual address is 192.168.1.1):

```
System.Execute("%windir%\system32\cmd.exe","%windir%\system32","/K ping 192.168.1.1");
```

System KeyDown / KeyUp / KeyPress

This functions allow to send a combination of 1 to 4 keys by a simple string describing this keyboard combination, like “CTRL+SHIFT + F10” or simply “0”.

```
System.KeyDown(szKey);
```

```
System.KeyUp(szKey);
```

Example:

```
System.KeyDown("A");
System.KeyDown("SHIFT+T");

System.KeyUp("A");
System.KeyUp("SHIFT+T");

System.KeyDown("CTRL+NP1");
System.KeyDown("ALT+F8");

System.KeyUp("CTRL+NP1");
System.KeyUp("ALT+F8");
```

KeyPress function send Down and UP message in a single function.

```
System.KeyPress("CTRL+NP1");
System.KeyPress("ALT+F8");
```

List of Key Name:

Regular Keys	NUM PAD	Special Key	FUNCTION
0 to 9	NP0 to NP9	BROWSERBACK	SHIFT
A to Z	NPMUL	BROWSERFORWARD	CTRL
BACK	NPADD	BROWSERREFRESH	ALT
TAB	NPDOT	BROWSERSTOP	
RETURN	NPSUB	BROWSERSEARCH	LWIN
ESC	NPDEC	BROWSERFAV	RWIN
SPACE	NPDIV	BROWSERHOME	LSHIFT
PAGEUP	NUMLOCK	VOLUMEMUTE	RSHIFT
PAGEDOWN	SCROLLLOCK	VOLUMEDOWN	LCTRL
END	CAPSLOCK	VOLUMEUP	RCTRL
HOME	PRINTSCREEN	MEDIANEXT	LMENU
LEFT	PAUSE	MEDIAPREV	RMENU
UP	CLEAR	MEDIASTOP	
RIGHT	SELECT	MEDIAPAUSE	F1 to F12
DOWN	PRINT	LAUNCHMAIL	F13 to F24
INSERT	PRINTSCREEN	MEDIASELECT	
DELETE	HELP	LAUNCHAPP1	
	APP	LAUNCHAPP2	
	EXECUTE	PLAY	