

MXwendler **OSC Interface Description Version 2.0**

This document describes the MXwendler (MXW) OSC Command Interface. You will learn how to control MXwendler through the OSC interface. We will describe the hierarchical control approach and how to assemble OSC command strings.

02.06

1. What is OSC?
 - 1.1 Why control MXW through OSC?
2. How do i receive OSC commands?
3. How do i send OSC from PD?
4. How do i send OSC from Max/MSP?
5. MXW OSC command structure
 - 5.1 MXW hierarchical command structure
 - 5.2 MXW OSC command set reference

1. What is OSC?

OSC stands for „Open Sound Control“. It was developed as a successor of the MIDI control. From the OpenSound Control home page1: „OpenSound Control ("OSC") is a protocol for communication among computers, sound synthesizers, and other multimedia devices that is optimized for modern networking technology“.

OSC Features

- Open-ended, dynamic,URL-style symbolic naming scheme
- Numeric and symbolic arguments to messages
- Pattern matching language to specify multiple targets of a single message
- High resolution time tags
- "Bundles" of messages whose effects must occur simultaneously
- Query system to dynamically find out the capabilities of an OSC server and get documentation

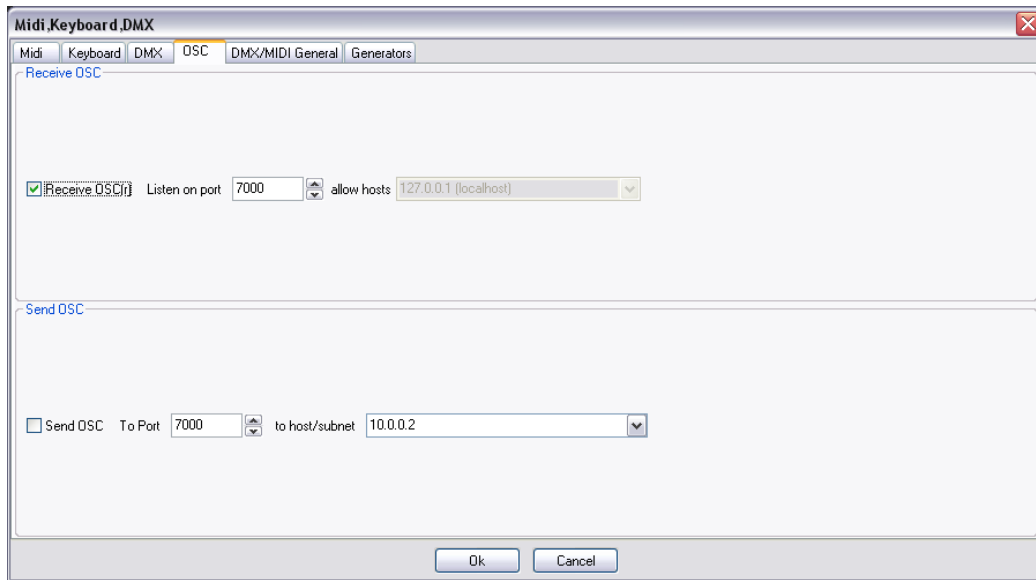
For more Information please see the OSC home page at the Berkeley University or the Internet Community Page opensoundcontrol.org.

1.1 Why control MXW through OSC?

MXW is designed to be a high-performance media server. It interactively composites high-resolution footage and interactive media like flash media. OSC commands can be generated and sent from a wide range of applications and signalling frameworks like pd, Max/MSP, eyesweb and many more. Separating the media servers from the controlling infrastructure gives you various tremendous creative options: create, control and sync panoramic scenes, separate scene analysis (eg. audio, video) from the compositing tasks for performance reasons, play distributed places, create custom interfaces for you or your customers or save money by seperating expensive content creation systems from cheap render slaves.

2. How do i receive OSC commands?

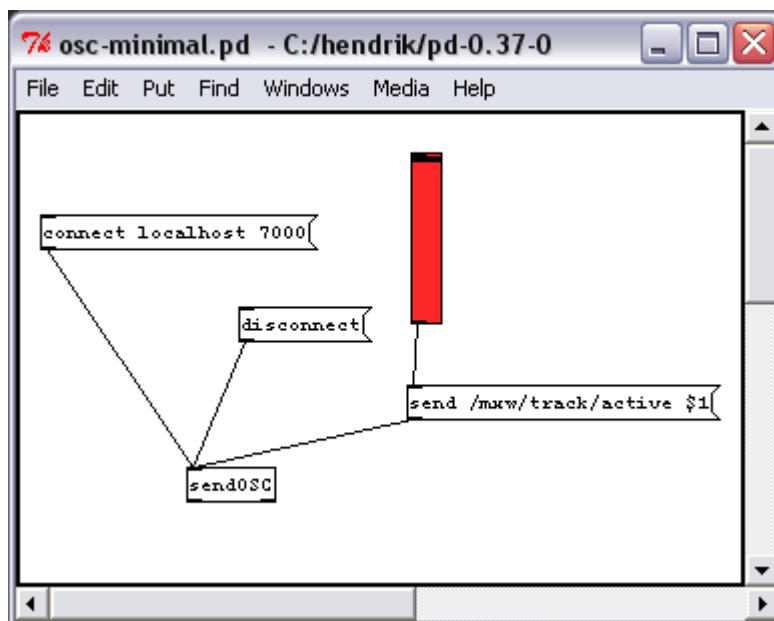
Enabling OSC receiving is very easy: Just turn it on by checking „Start OSC“ in the Open Sound Control Panel (Settings -> Midi,DMX,Keyboard,OSC):



You have to restart your application to apply any changes. Starting OSC will open a listening network port, so you should be aware about this potential security risk. You can restrict listening to certain IP addresses (eg. 127.0.0.1) or subclasses, but there is no guarantee that this restriction will secure your system. In our example we will try to connect from a PD instance on our machine, so you should restrict listening to the localhost IP address 127.0.0.1. Do NOT turn on OSC sending (by now).

3. How do i send OSC commands from PD?

You need a PD with a running sendOSC object. A minimal patch looks like this:

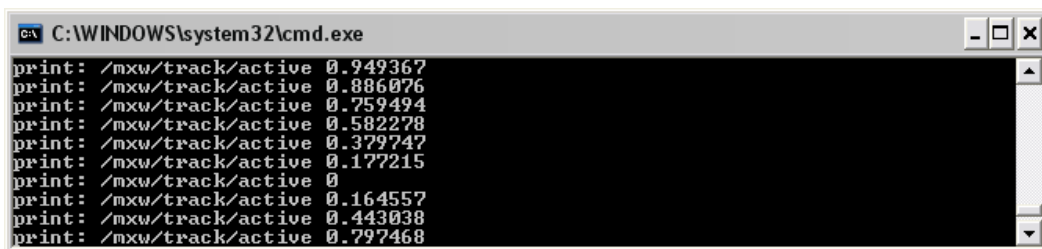


After you click on [connect localhost 7000] you should see „connected to port localhost:7000“ in your command window. If you see something else, this port is either already in use or you are blocked by a firewall.

To send an OSC command, you have to send a string with a special syntax. To put the MXW main fader in the middle position, create a patch like the patch above:

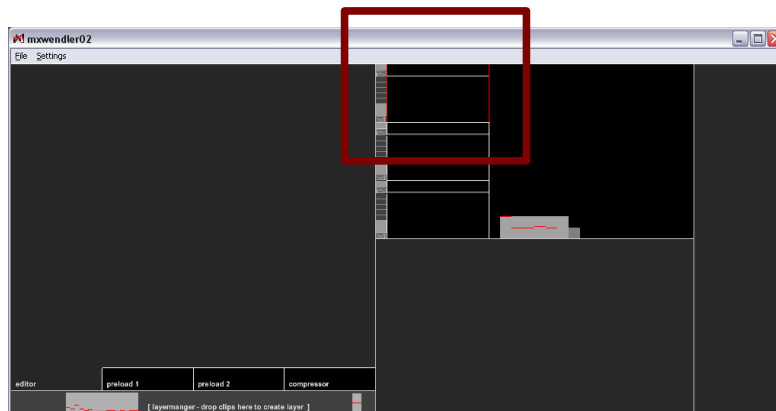
1. enter edit mode in pd (ctrl-e)
2. create a sendOSC object
2. format OSC message: „send /mxw/track/active \$1“
3. connect a slider to the messagebox, bottom value 0.0 top value 1.0
4. connect the messagebox to the sendOSC object
5. leave edit mode and drag the slider

Thats all. Now, when you drag the slider, you should see messages like these in the pd command window:



```
C:\WINDOWS\system32\cmd.exe
print: /mxw/track/active 0.949367
print: /mxw/track/active 0.886076
print: /mxw/track/active 0.759494
print: /mxw/track/active 0.582278
print: /mxw/track/active 0.379747
print: /mxw/track/active 0.177215
print: /mxw/track/active 0
print: /mxw/track/active 0.164557
print: /mxw/track/active 0.443038
print: /mxw/track/active 0.797468
```

If you did everything right, you should see the active track wipe status following your pd slider:



4. MXW OSC command structure

Commands to the MXW OSC interface always follow the same syntax:

/mxw/[address of widget] [floatval from 0..1|string]

Commands always begin with „/mxw/“. The address of the widget is determined by a hierarchical access beginning with the tracks, continuing over the layers descending to the clips. The command receivers listen either to a float (if they are widgets) or to strings (to choose eg. patches or clips). Float values are always in the range 0.0 .. 1.0 and will be scaled to the actual ranges. Eg. the clip playback rate can be set in a range from -5.0 .. 5.0 by dragging from the slider from bottom to top -the according OSC values are nevertheless 0.0 .. 1.0. This behaviour is designed to be logically

as close as possible to Midi ranges, which are always from 0 .. 127 and will be scaled to the actual useful ranges as well.

To address the main fader and fade from 0 .. 1 in 10 steps, you would send a command sequence

```
/mxw/render 0.0
/mxw/render 0.1
/mxw/render 0.2
...
/mxw/render 1.0
```

You are free to send finer grained values. The commands will be evaluated as soon as they arrive. The last command before a new frame is drawn will set the widgets value. There is no need the send commands more often than the MXWendler renderer updates.

5. MXW hierarchical command addressing scheme

Commands to the MXW OSC interface follow a hierarchical system, no matter if you use OSC to access MXW or MIDI or your keyboard. To address a certain widget, use always the following system:

```
"/mxw/set", // choose a patch by sending 0..1 OR 1|2|3 ...
"/mxw/trackmanager", // choose active track
"/mxw/layermanager", // choose active layer
"/mxw/render", // send value to active track (main fader)
"/mxw/render/opacity", // change of render opacity
"/mxw/render/scale", // ..
"/mxw/render/scalexy", // ..
"/mxw/render/translationx", // ..
"/mxw/render/translationy", // ..
"/mxw/render/rotation", // ..
"/mxw/render/aspectratio", // ..
"/mxw/render/reset", // ..

"/mxw/track/1", // send value to specific track
"/mxw/track/active", // like render

"/mxw/preload/1/trigger", // trigger preload 1 content to current track
"/mxw/preload/2/trigger", // ..
"/mxw/preload/10/trigger", // ..

"/mxw/preload/1/flipflop", // flipflop preload 1 content to current track
"/mxw/preload/2/flipflop", // ..
"/mxw/preload/3/flipflop", // ..

"/mxw/track/active/layer/1", // adress of specific layer (does nothing)
"/mxw/track/active/layer/active", // adress of active layer
"/mxw/track/active/layer/active/opacity", // change opacity of active layer,track
"/mxw/track/active/layer/active/scale", // ..
"/mxw/track/active/layer/active/scalexy", // ..
"/mxw/track/active/layer/active/translationx", // ..
"/mxw/track/active/layer/active/translationy", // ..
"/mxw/track/active/layer/active/rotation", // ..
"/mxw/track/active/layer/active/mode", // ..
"/mxw/track/active/layer/active/aspectratio", // ..
"/mxw/track/active/layer/active/reset", // ..
```

```

"/mxw/track/active/layer/active/clip/keyin", // send to key in of clip of active layer,track
"/mxw/track/active/layer/active/clip/keyout", // ..
"/mxw/track/active/layer/active/clip/speed", // ..
"/mxw/track/active/layer/active/clip/position", // ..
"/mxw/track/active/layer/active/clip/speed", // ..
"/mxw/track/active/layer/active/clip/mode", // ..
"/mxw/track/active/layer/active/clip/reset", // ..
"/mxw/track/active/layer/active/clip/effect", // ..
"/mxw/track/active/layer/active/clip/moreeffects", // create effect. [delete effect] is missing
"/mxw/track/active/layer/active/clip/effectbatch", // choose active effect
"/mxw/track/active/layer/active/clip/clips", // choose from clip library

"/mxw/track/active/layer/active/clip/effect/1/param/1", // set param 1 of effect 1
"/mxw/track/active/layer/active/clip/effect/1/param/2",
"/mxw/track/active/layer/active/clip/effect/2/param/1",
"/mxw/track/active/layer/active/clip/effect/2/param/2"

```

Examples:

```
/mxw/track/active 0.5
```

Fades the active track to 0.5

```
/mxw/preload/1 add
```

Adds a layer with the contents of preload slot 1

```
/mxw/track/1/layer/0 opacity 0.5
```

Sets the opacity of the first layer of the lower track to 0.5

```
/mxw/set 5
```

The active track will load patch 5

```
/mxw/track/active/layer/active/clip speed 1.0
```

Sets the speed of the clip of the active layer of the active track to 1.0 (5x fast forward)

```
/mxw/trackmanager 0.2
```

Activate the first track

```
/mxw/trackmanger 0.7
```

Activate the third track

```
/mxw/track/2/layer/1/clip effect 0.1
```

Instructs the clip of the second layer of the upper track to load the first of ten effects

```
/mxw/track/2/layer/0/clip/effect/0/param/0 set 0.8
```

Sets the effect parameter 0 of the first effect of the clip of the first layer of the upper track to the value 0.8

- 1 <http://www.cnmat.berkeley.edu/OpenSoundControl/>
- 2 A very good OSC resource: <http://www.opensoundcontrol.org>

