

i-score: an overview

Linux Audio Conference - Workshop

Jean-Michaël Celerier

May 19, 2017

Context

Foundation: libossia

Goal

Protocols

Interoperability

The sequencer: i-score

Control

Temporal structure

Interactivity

Devices

Audio features

Conclusion

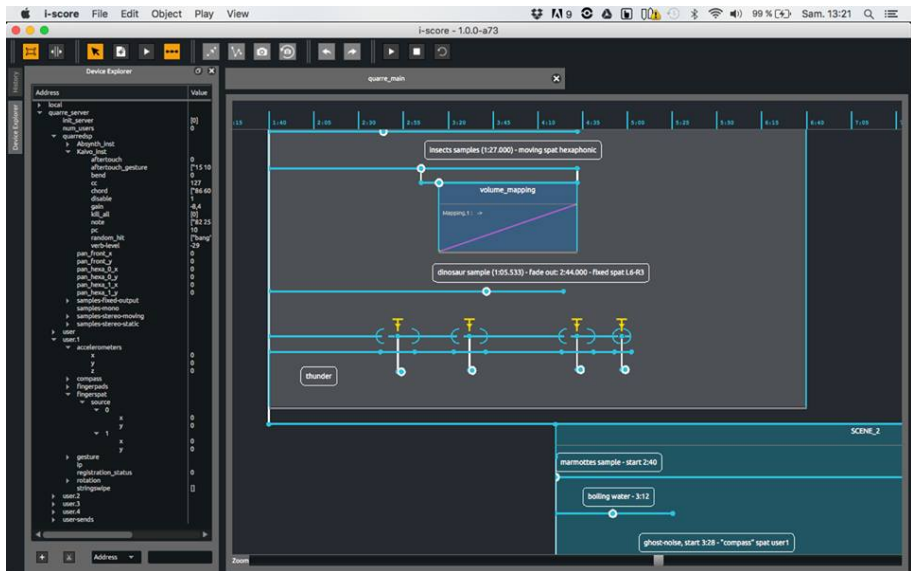
Workshop

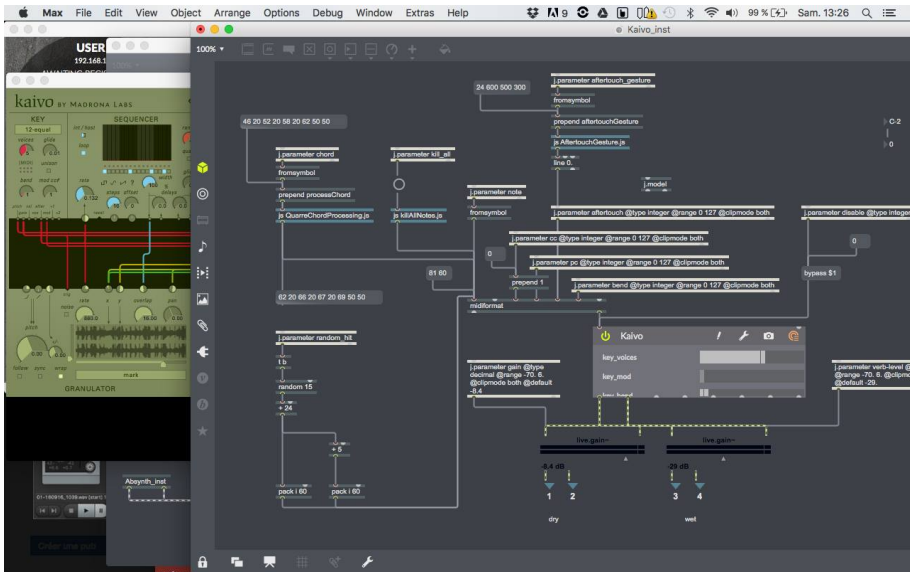
Les Baltazars - Tumbleweed



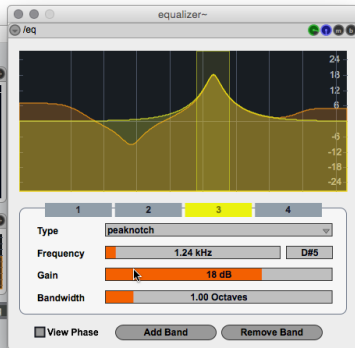
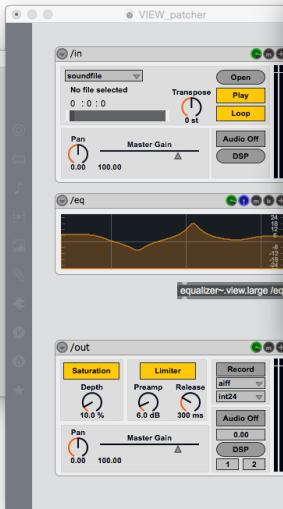
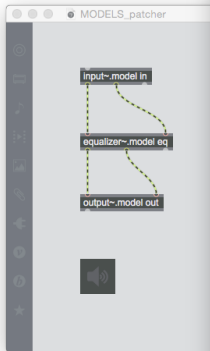
Pierre Cochard - Quarre







- ▶ Digital arts: music, video, transmedia, etc...
- ▶ Temporal structure & interactivity.
- ▶ Interoperability: software, hardware.



libossia: goals

- ▶ Automatic discovery.
- ▶ Shared object model inspired from OSC.
- ▶ Scoring primitives.

libossia: protocols

- ▶ OSC
- ▶ Minuit
- ▶ OSCQuery
- ▶ MIDI
- ▶ HTTP (Requires Qt)
- ▶ WebSocket (Requires Qt)
- ▶ Serial port (Requires Qt)

To come: ArtNet (DMX)

Standard protocols

Address	Value	Get	Set	Min	Max
▼ OSCdevice					
▼ mouse					
move	[0, 0]	✓	✓		
click	[0, 0]	✓	✓		
release	[0, 0]	✓	✓		
▼ particle					
density	0	✓	✓	1	50
radius	0	✓	✓		
color	[0, 0, 0]	✓	✓		
▼ VDMX					
▼ layer.1					
alpha	1	✓	✓	0	1
▼ position					
x	0.5	✓	✓	0	1
y	0.5	✓	✓	0	1
▼ scale					
x	0.5	✓	✓	0	1
y	0.5	✓	✓	0	1

Qt-based protocols

When no tree structure easily makes sense,
let the user script it !

```
function onMessage(message) {
    console.log(message);
    var res = JSON.parse(message);
    console.log(res.value);
    if(res.name === "toto")
    {
        return [ { address: "/toto", value: res.value } ];
    }
    return { };
}

function createTree() {
    return [ {
        name: "tata",
        children: [
            {
                name: "tutu",
                request: "{ \"name\": \"toto\", \"value\": $val",
                type: Ossia.Float,
                unit: "color.rgb"
            },

```

libossia (C++14)

Linux, macOS, Windows, GCC, Clang, MSVC, static, dynamic...
Only header-only dependencies.

```
auto& node = find_or_create_node(device, "/test/my_int");
auto address = node.create_address(val_type::INT);

node.set(access_mode_attribute{}, access_mode::GET);
node.set(bounding_mode_attribute{}, bounding_mode::FOLD);
node.set(domain_attribute{}, make_domain(2, 14));
node.set(description_attribute{}, "an integral value");

address->add_callback([] (const auto& val) {
    std::cout << val << " ";
});

address->push_value(5678);
```

Integration with ofParameter, ofParameterGroup

```
ossia::Parameter<bool> _fill;  
ossia::Parameter<ofColor> _color;  
ossia::ParameterGroup _sizeParams;  
...  
_circleParams.setup(_parent_node, "circle");  
  
_sizeParams.setup(_circleParams, "sizeParams");  
_radius.setup(_sizeParams, "radius", 10., 1., 100.);  
_position.setup(_sizeParams,  
    "position",  
    ofVec2f(ofGetWidth() / 2, ofGetHeight() / 2),  
    ofVec2f(0., 0.), // Min  
    ofVec2f(ofGetWidth(), ofGetHeight())); // Max
```


ossia-pd (and ossia-max soon)

```
ossia.param note @type vec2f @description "Midi note and velocity"
```

```
t a a
```

```
stripnote $2
```

```
mtof
```

```
0
```

```
ossia.param modratio @default 4 @type float @description  
"Modulator frequency ratio"
```

```
ossia.param modgain @default 0 @type float @description  
"Modulator frequency gain" @unit dB
```

convert dBFs to Pd's dB, 100dB Pd = 0dBFS

```
4
```

```
* 4
```

```
0
```

```
+ 100
```

```
dbtorms
```

```
$1 10
```

```
line~
```

```
phasor~
```

```
*~
```

```
osc~
```

```
*~
```

```
loadbang
```

```
;pd dsp 1
```

```
loadbang
```

```
dump
```

```
expose oscquery
```

```
ossia.device Pd
```

```
print
```

ossia-python

```
# create a node, create a tuple address and initialize it
tuple_node = local_device.add_node("/test/value/tuple")
tuple_address = tuple_node.create_address(
    ossia.ValueType.Tuple)
tuple_value = ossia.Value([
    ossia.Value(44100),
    ossia.Value("test.wav"),
    ossia.Value(0.9)]
)
tuple_address.push_value(tuple_value)

# attach a callback function to the boolean address
def bool_value_callback(v):
    print(v.get())
    bool_address.add_callback(bool_value_callback)
```

ossia-unity3D (C#)

```
public class Foo : public MonoBehaviour
{
    [Ossia.Attribute]
    int foo;
}
```

ossia-qml (Qt QML)

```
Item {  
    Ossia.Node { name: 'test' }  
    AngleSlider {  
        // Reads and writes from /test/angle  
        Ossia.Property on angle {  
            min: -90  
            max: 0  
            bounding: Ossia.Context.Clip  
        }  
    }  
}
```

ossia-C (C99)

```
OSSIA_EXPORT
bool ossia_device_update_namespace(
    ossia_device_t device);

OSSIA_EXPORT
ossia_node_t ossia_device_get_root_node(
    ossia_device_t device);

OSSIA_EXPORT
const char* ossia_device_get_name(
    ossia_device_t node);

///// Node /////
OSSIA_EXPORT
ossia_node_t ossia_node_add_child(
    ossia_node_t node,
    const char * name);
```

Demonstration

i-score + PureData + Processing

Sending messages

The screenshot displays a software interface for managing messages, likely for a hardware device. The interface is divided into several sections:

- Top Bar:** Contains menu items (File, Edit, Object, Play, Tool, View, Settings, About) and a toolbar with icons for various functions.
- Device Explorer:** Located on the left, it shows a tree view with the following structure:

Address	Val
OSCdevice	
- Timeline Editor:** The central area features a horizontal timeline with time markers at 0:00.0, 0:02.500, 0:05.0, 0:07.500, and 0:10.0. A blue line labeled "Example /" spans from 0:00.0 to 0:10.0. A blue circle with a yellow star is positioned on the timeline at approximately 0:02.500.
- Inspector:** Located on the right, it displays properties for the selected object, "laws73dock5". The properties include:
 - Name:** laws73dock5
 - Label:**
 - Default date:** 2 s 234 ms
 - Trigger:** Enable trigger
 - Events:**
 - rash75oval66:** Name: rash75oval66, Label: Parent TimeNode
 - Condition:**
 - State:** A table showing the current state of the device:

Address	Value
OSCdevice example	0
 - Prev. Constraint:** Add process

Automating

The screenshot displays a software interface for automating OSC device control, featuring a timeline, a device explorer, and an inspector panel.

Device Explorer: A table listing OSC devices and their values.

Address	Val
OSCdevice	
example	0

Timeline: A horizontal timeline with a time scale from 0:00.0 to 0:10.0. A blue line labeled "Example /" spans the duration. A blue box labeled "automation" is positioned on the timeline, containing the text "...Ice/example".

Inspector: A panel on the right side of the interface showing the properties of the selected object.

Name: automation

Label:

Speed x1: x0 x0.5 x1 x2 x5

Start State: Full view

End State:

Default Duration: 0.00.03.786

Add Process:

Automation.1:

Address: OSCdevice/example

Tween: ☐

Min: 0.00000

Max: 1.00000

Unit: None

Start: OSCdevice/example 0

End: OSCdevice/example 1

Interpolating

The screenshot shows a software interface for creating an automation clip. The main workspace displays a timeline with a clip labeled "Example /" and an automation clip labeled "automation" containing the text "...Ice/example". The timeline has markers at 0:00.0, 0:02.500, 0:05.0, 0:07.500, and 0:10.0. The automation clip is a blue rectangle with a green line indicating a linear interpolation from a value of -2 at the start to 10 at the end.

Device Explorer

Address	Val
OSCdevice	
example	10

Inspector

Name automation

Label

Speed x1
x 0 | x 0.5 | x 1 | x 2 | x 5

Start State Full view End State

Default Duration 0.00.03.786

+ Add Process

Interpolation.1

Address OSCdevice/example

Tween ☐

Start -2

End 10

Tweening

The screenshot displays a software interface for creating and editing tweens. The interface is divided into several panels:

- Top Panel:** Contains a menu bar (File, Edit, Object, Play, Tool, View, Settings, About) and a toolbar with various icons for navigation and editing.
- Device Explorer:** Located on the left, it shows a table with columns for Address and Value. The table is currently empty.
- Timeline:** A horizontal timeline at the top of the main workspace with markers at 0:00.0, 0:02.500, 0:05.0, 0:07.500, and 0:10.0. A blue line represents the timeline, and a yellow arrow points to the 0:10.0 mark.
- Graph:** A graph area below the timeline. It shows a blue rectangular area labeled "tweening" and a red line graph with two points, indicating a value change over time.
- Inspector:** Located on the right, it displays the properties of the selected tween. The "Name" field is set to "tweening". The "Speed" is set to "x1". The "Default Duration" is 0.00.06.953. The "Automation.1" section shows the "Address" field, a "Tween" checkbox, and "Min" and "Max" values set to 0.00000. The "Unit" is set to "None". The "Start" and "End" values are both 0.

Mapping

The screenshot displays a software interface for mapping OSC devices, featuring a menu bar, a Device Explorer, a central timeline, and an Inspector.

Menu Bar: File, Edit, Object, Play, Tool, View, Settings, About.

Device Explorer: A table showing the hierarchy of OSC devices.

Address	Val
OSCdevice	
other	0
example	0

Central Timeline: A horizontal timeline with time markers at 0:00.0, 0:02.500, 0:05.0, 0:07.500, and 0:10.0. A blue box labeled "mapping" is positioned over the timeline, containing a graph with a purple curve. The graph is labeled "Mapping.1: ...ice:/example -> ...vice:/other".

Inspector: A panel on the right showing the properties of the selected object, "mapping".

Name: mapping

Label:

Speed x1: x 0, x 0.5, x 1, x 2, x 5

Buttons: Start State, Full view, End State

Default Duration: 0.00.05.525

Add Process: + Add Process

Mapping.1:

Source: OSCdevice:/example

Min: 0.00000

Max: 1.00000

Target: OSCdevice:/other

Min: -100.00000

Max: 100.00000

Scripting

The screenshot displays a software interface with a dark theme, organized into several panels:

- Top Panel:** Contains a menu bar with 'File', 'Edit', 'Object', 'Play', 'Tool', 'View', 'Settings', and 'About'. Below the menu is a toolbar with icons for various functions like zooming, panning, and execution.
- Device Explorer (Left Panel):** A panel with a table showing 'Address' and 'Val' columns. It includes a search bar and a list of addresses.
- Timeline (Center):** A horizontal timeline with markers at 0:00.0, 0:02.500, 0:05.0, 0:07.500, and 0:10.0. A blue bar labeled 'Example /' spans the duration. Below it, a 'script' block is shown with a code editor containing the following JavaScript code:

```
1 (function(t) {  
2   var obj = new Object;  
3   obj['address'] = 'OSCdevice:/millumin/layer/x/instance';  
4   obj['value'] = t + iscore.value('OSCdevice:/millumin/layer/y/instance');  
5   return [ obj ];  
6 });
```
- Inspector (Right Panel):** A panel for configuring objects. It shows a 'Name' field with 'Example' and a 'Label' field. Below these are 'Speed x1' controls with a range from x0 to x5. A 'Trigger' dropdown is set to 'False'. There are buttons for 'Start State', 'Full view', and 'End State'. At the bottom, there are settings for 'Default Duration' (0.00.10.000), 'Min Duration' (0.00.10.000), and 'Max Duration' (Infinity). A 'Scenario.1' dropdown is also visible.

Racks

The screenshot displays the Racks software interface, which is used for creating and editing audio racks and automation.

Top Bar: Contains standard menu items (File, Edit, Object, Play, Tool, View, Settings, About) and a toolbar with icons for various functions like zoom, pan, and playback.

Device Explorer: Located on the left, it shows a hierarchy of devices. The current selection is 'example' under 'OSCdevice'.

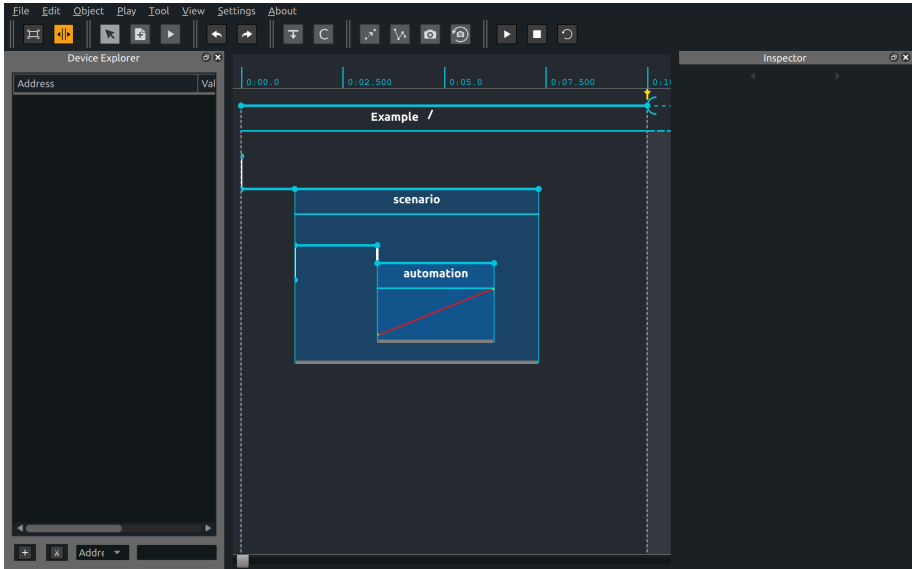
Timeline: The central area shows a timeline with a blue line representing the rack's duration. The timeline is divided into segments with time markers: 0:00.0, 0:02.500, 0:05.0, 0:07.500, and 0:10.0. A label 'Example /' is visible above the timeline.

Rack: A blue rectangular area containing a mapping table. The mapping table is titled 'Mapping.1' and shows a mapping from 'OSCdevice/example' to 'OSCdevice/other'. The mapping is visualized with a red line graph showing the transition over time.

Inspector: Located on the right, it provides detailed settings for the selected object. The current object is 'Name rack'. The inspector shows the following settings:

- Name:** rack
- Label:** Label
- Speed:** x1 (with a slider and buttons for x0, x0.5, x1, x2, x5)
- Start State:** Start State (button)
- Full view:** Full view (button)
- End State:** End State (button)
- Default Duration:** 0.00.05.525
- Add Process:** Add Process (button)
- Mapping.1:**
 - Source:** OSCdevice/example
 - Min:** 0.00000
 - Max:** 1.00000
 - Target:** OSCdevice/other
 - Min:** -100.00000
 - Max:** 100.00000
- Automation.2:**
 - Address:** Address
 - Tween:** ☐
 - Min:** 0.00000
 - Max:** 0.00000
 - Unit:** None
 - Start:** 0
 - End:** 0
- Loop.3:** Loop.3 (button)
- Name:** pattern

Hierarchy



Loops

The screenshot displays a software interface for creating and editing loops and patterns. The main workspace shows a timeline with a blue bar labeled "Example /" and a nested blue bar labeled "loop". Inside the "loop" bar is a purple bar labeled "pattern". A red line connects the start of the "pattern" bar to the end of the "loop" bar, indicating a loop connection. The timeline has time markers at 0:00.0, 0:02.500, 0:05.0, and 0:07.500.

The interface includes a menu bar (File, Edit, Object, Play, Tool, View, Settings, About) and a toolbar with various icons. On the left is a "Device Explorer" panel with "Address" and "Value" columns. On the right is an "Inspector" panel showing the properties of the selected object, "loop".

The "Inspector" panel for "loop" shows:

- Name: loop
- Label: (empty)
- Speed x1: x0 | x0.5 | x1 | x2 | x5
- Buttons: Start State, Full view, End State
- Default Duration: 0.00.07.017
- + Add Process
- Loop.1
 - Name: pattern
 - Label: (empty)
 - Speed x1: x0 | x0.5 | x1 | x2 | x5
 - Buttons: Start State, Full view, End State
 - Default Duration: 0.00.03.437
 - + Add Process
 - Automation.1
 - Address: (empty)
 - Tween: ☐
 - Min: 0.000000
 - Max: 0.000000
 - Unit: None
 - Start: 0
 - End: 0

Interactivity: conditions

The screenshot displays a game engine interface with a timeline and an Inspector panel.

Timeline: The timeline shows a sequence of events. A blue line represents a condition node labeled "Condition /". A yellow circle highlights a specific point on the timeline, and a yellow arrow points to it. The timeline has markers at 0:00.0, 0:02.500, 0:05.0, 0:07.500, and 0:10.0.

Inspector: The Inspector panel shows the properties of the selected object, "bibb54hymn24". The properties include:

- Name: bibb54hymn24
- Label
- Default date: 4 s 809 ms
- Trigger: Enable trigger
- Events
 - skim8bogs97
 - alto6axon42
 - Name: alto6axon42
 - Label
 - Parent TimeNode
 - Condition
 - a/b \geq 3
 - Offset behaviour: True
 - State
 - State
 - flex20rung34

Interactivity: trigger points

The screenshot displays a software interface for managing trigger points. The main area is a timeline labeled "Example /" with a time axis from 0:00.0 to 0:07.500. The timeline features several horizontal blue lines representing processes, with yellow trigger points (marked with a 'T' and a plus sign) indicating specific events. These events are connected to various processes, some of which are shown as dashed lines, suggesting they are not yet fully defined or are placeholders.

On the left, the "Device Explorer" panel shows a table with columns "Address" and "Val".

On the right, the "Inspector" panel displays the properties of the selected object, "rips75scoot16". The properties include:

- Name: rips75scoot16
- Label
- Default date: 3 s 869 ms
- Trigger: foo/bar, ≥, 3
- Events
 - grim88joke98
 - Name: grim88joke98
 - Label
 - Parent TimeNode
- Condition
 - State
 - Address
 - Value
- Prev_Constraint
- Add process

Interactivity: execution speed

The screenshot displays a software interface for managing execution speed, likely for a game engine or animation tool. The interface is divided into several panels:

- Top Panel:** Contains a menu bar (File, Edit, Object, Play, Tool, View, Settings, About) and a toolbar with various icons for navigation and editing.
- Device Explorer (Left Panel):** A panel with a table showing device information. The table has two columns: "Address" and "Val".
- Timeline (Center Panel):** A horizontal timeline with a vertical playhead. The timeline is divided into segments labeled "0:00.0", "0:10.0", "0:20.0", and "0:30.0". A blue line represents the main timeline, and a green line represents a sub-timeline. A yellow playhead is positioned at the end of the timeline. A label "Example /" is visible above the timeline.
- Inspector (Right Panel):** A panel showing the properties of the selected object. It includes a "Name" field (jilt95bbs52), a "Label" field, and a "Speed x5" dropdown menu with options "x0", "x0.5", "x1", "x2", and "x5". Below the speed dropdown are buttons for "Start State", "Full view", and "End State". A "Default Duration" field is set to "0.00.33.259". At the bottom, there is a button labeled "+ Add Process".

Working with external devices

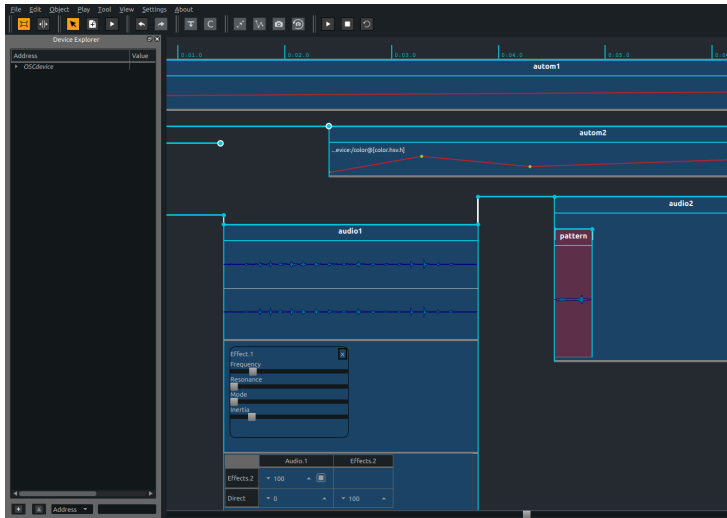
- ▶ Manual entry
- ▶ Loading
- ▶ Learning
- ▶ Automatic discovery

Demonstration

MIDI control surface and WebSockets



Audio

Hierarchical mixing, sounds, effects (Faust, LV2), sends...



The whole audio sequencing part is implemented with the plug-in API.

What's missing

- ▶ Multichannel operation.
- ▶ Displaying LV2 UIs...
- ▶ Musical time structures (bars, metronome, etc).
- ▶  Packaging for distros 

Work-in-progress

- ▶ Embedded score player: put your score in an Android app, a Pd patch, ...
- ▶ Network edition, execution and control, of both editors and players.
- ▶ Full-fledged audiograph.
- ▶ Ongoing work on UI.
- ▶ Need to test live edition more.

Workshop

- ▶ Building scores.
- ▶ Experimenting with your favorite environments.
- ▶ Gather your remarks and advices !