



# Interoperability/Reusability of high level WebAudio components

Michel Buffa

## ► To cite this version:

Michel Buffa. Interoperability/Reusability of high level WebAudio components. W3C Workshop on Web Games, Jun 2019, Redmond, United States. hal-02366763

**HAL Id: hal-02366763**

**<https://inria.hal.science/hal-02366763>**

Submitted on 16 Nov 2019

**HAL** is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

# Interoperability/Reusability of high level WebAudio components

Games on Web W3C Workshop, Seattle 2019

Michel Buffa

Université Côte d'Azur

France

I3S/CNRS/INRIA labs

buffa@i3s.unice.fr

@micbuffa

WAMs

UNIVERSITÉ  
CÔTE D'AZUR

i3s  
sophie antipolis

inria

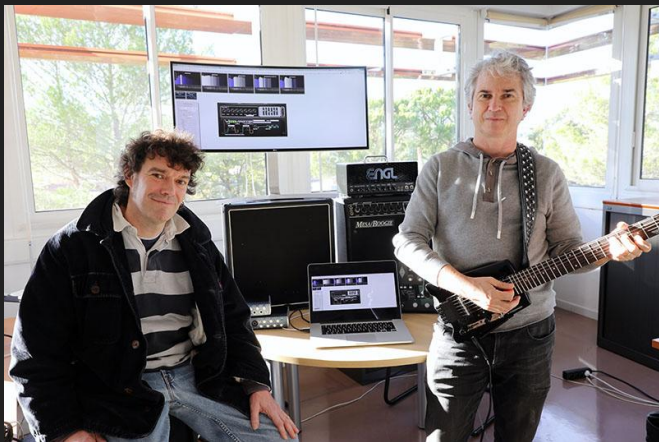
cnrs

wim  
mic

crame  
CENTRE NATIONAL DE RECHERCHE  
MUSICALE

# Who am I?

- Professor / researcher at Université Côte d'Azur, in the South of France
  - member of the WIMMICS research group common to INRIA and I3S lab from CNRS
- National coordinator of the WASABI ANR research project, with WebAudio at its heart,
- W3C Advisory Committee Representative for UCA
- I participate to the WebAudio working group



# Real-Time Emulation of a Marshall JCM 800 Guitar Tube Amplifier, Audio FX Pedals, in a Virtual Pedal Board

Michel Buffa, Jérôme Lebrun  
Université Côte d'Azur  
CNRS, INRIA  
(buffa, lebrun)@3s.unice.fr

- WebAudio / WebMidi APIs
- proposal for an open plugin standard (see VST2/3 for the Web)
- Targets: music schools, composers, ...



Guitar amp schematics

- Plugins can be:
- JavaScript classes
  - WebComponents (GUI)
  - WebAssembly (binary)
  - C/C++ compiled
  - FAUST compiled
- Support for local / remote plugins  
Integrated MIDI support  
Plugins by [webaudiomodules.org](http://webaudiomodules.org)  
FAUST DSL: <http://faust.greyc.fr>



Tube guitar amp simulator/designer, final amps packaged as plugins



Host Webapp for assembling amps/effects and instrument plugins



ANR WASABI  
<http://wasabihome.3s.unice.fr>





# Some ambitious WebAudio examples...

- AudioGraphs of these apps use high-level WebAudio nodes and / or AudioWorklets + WebAssembly.



But... no plugin standard, no “hosts”, no programming model...

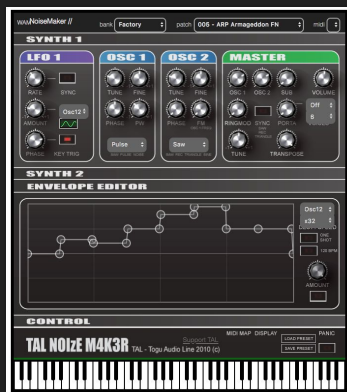
We find some very good JavaScript libraries (i.e. [toneJS](https://tonejs.github.io/))

Some open source github repositories (i.e. <https://webaudiodemos.appspot.com/>)

Some online tools for music synthesis ([genish.js](https://genish.js.org/) etc.)

Some DSL for DSP programming ([FAUST](https://faust.cc/), etc.)

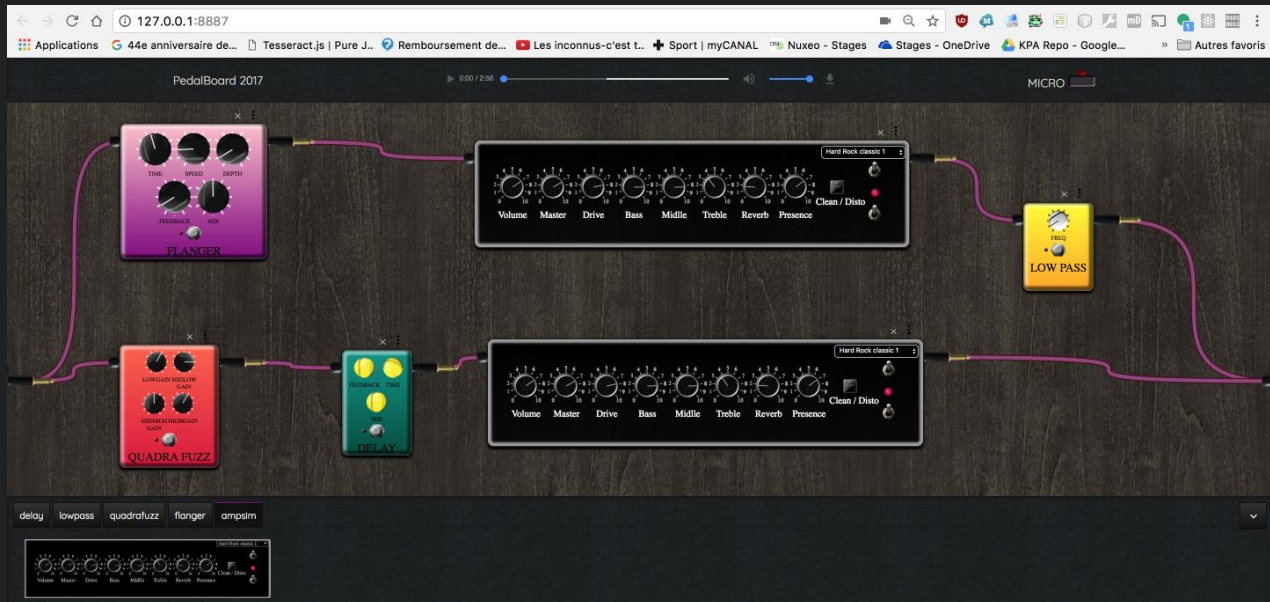
Some effects and instruments







In early 2018, with some researchers and developers we decided to start working on an open plugin standard for WebAudio





We made a team with different researchers / developers, that share same concerns with different approaches

- **1 - Bringing native developers to the Web**

- a. **Jari Kleimola** (Aalto University Espoo, Southern Finland, now at Webaudiomodules.org),
- b. **Oli Larkin** (Developer of VirtualCZ, Endless Series, WDL-OL/iPlug, iPlug2)

- **2 - Bringing low level DSP developers to the Web**

- a. **Stéphane Letz** (senior researcher at GRAME, Lyon, co-author of the FAUST DSL/compiler)

- **3 - Attract Web Developers / JavaScript audio app developers**

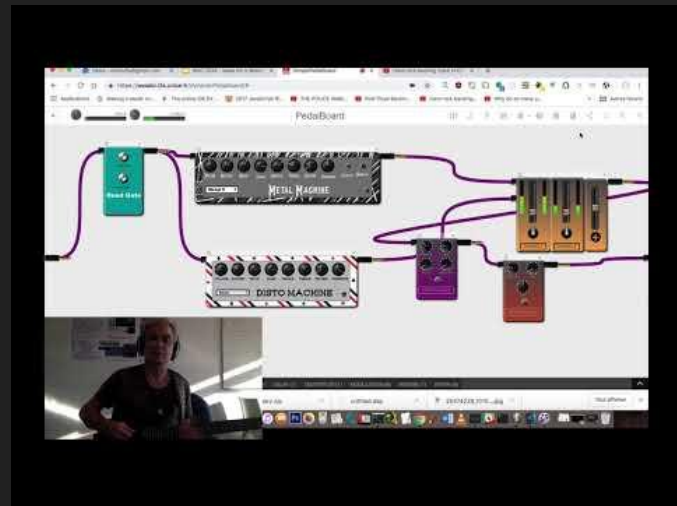
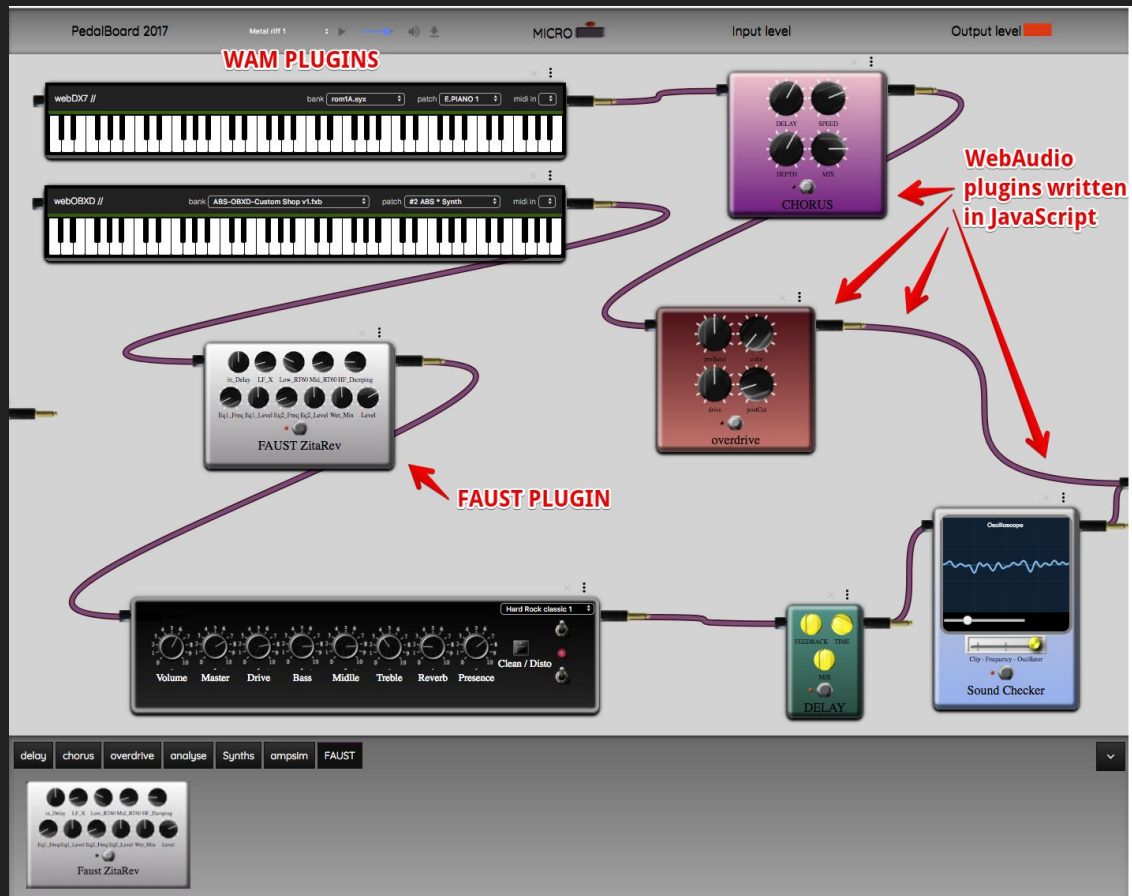
- a. **Tatsuya Shinyagaito, aka g200kg**, (Audio and WebAudio developer, huge WebAudio contributor, Yokohama, Kanagawa, Japan)
- b. **Jerôme Lebrun and Michel Buffa** (I3S/INRIA)

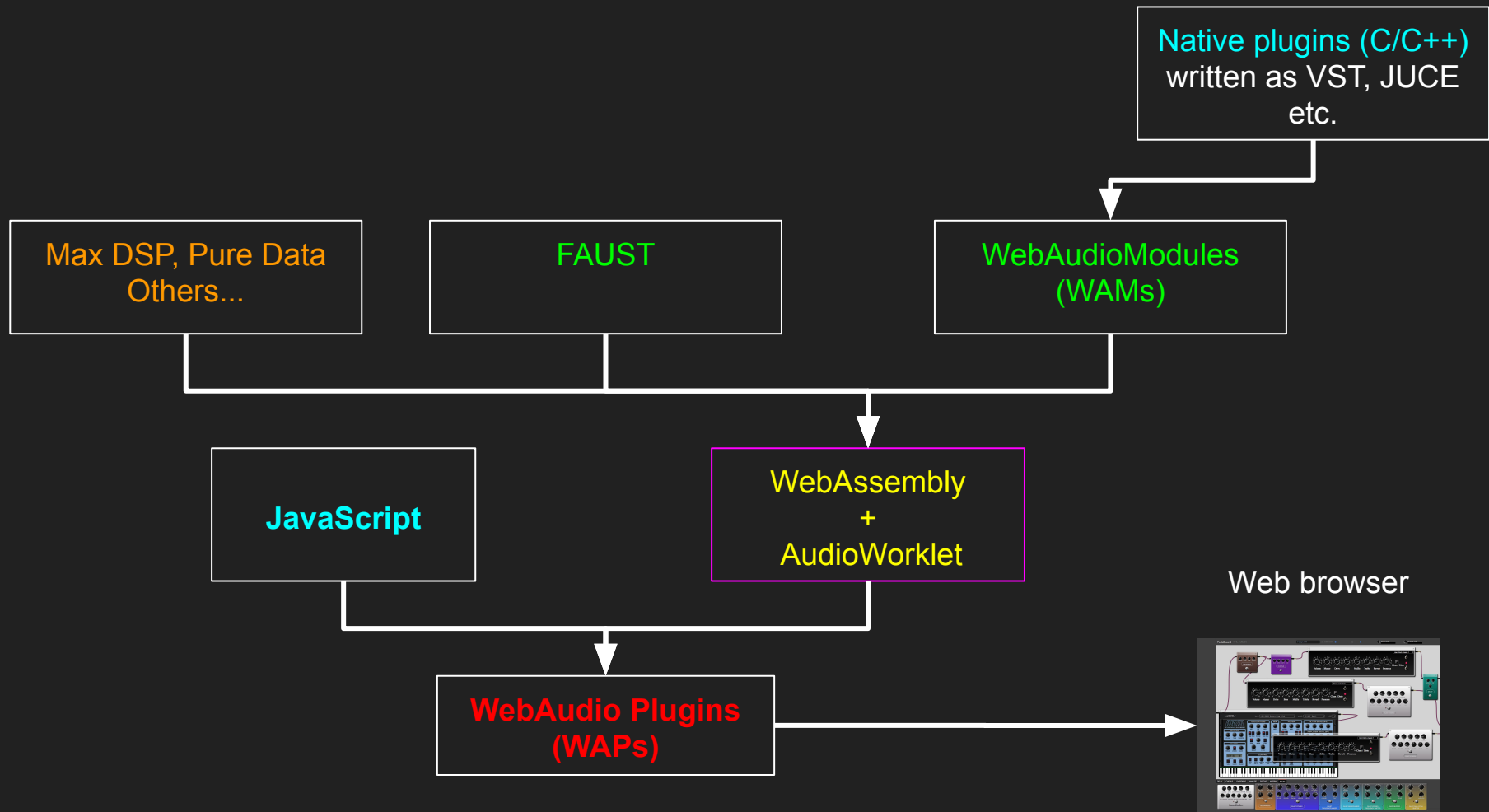
# An open standard = API/specification ? Or more... ?

## Be Web-Aware!

- **Use URIs** : support local or remote plugins, audio and midi routing
- **Asynchronous events in the lifecycle**
- **Plugins can be headless or with a GUI**
- **API for the “audio processor part”, as close as possible to AudioNode**
  - i.e use `plugin.connect(gain)` or `gain.disconnect(plugin)`, etc.
- **Propose an API or at least guidelines on how to package and publish plugins on REST repositories**
- **Avoid naming conflicts** (HTML ids, JS names, CSS rules), metadata...

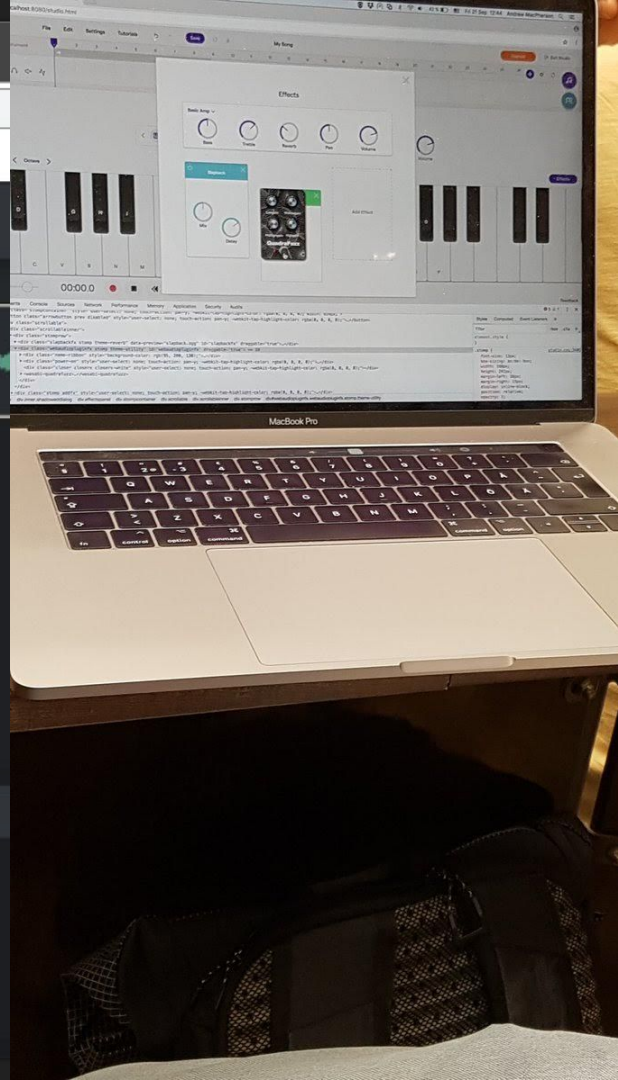
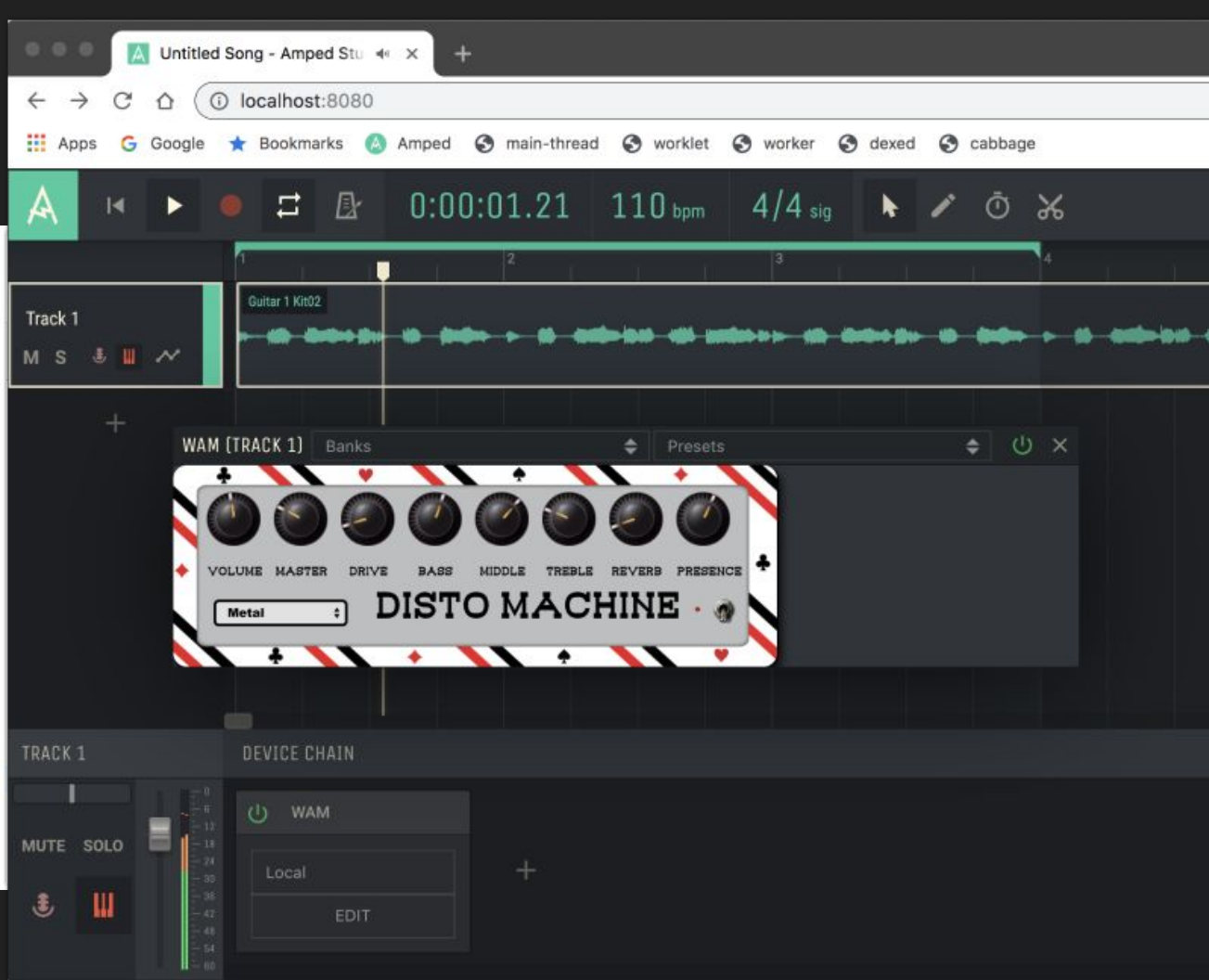
# Let's play with some WAPs (WebAudio Plugins)











# Low level DSP automatized -> WASM

The screenshot displays the Faust editor web interface, which is used for creating digital signal processing (DSP) code and its graphical user interface (GUI).

**Left Panel (Controls):**

- Run** button
- Share** button
- Poly Voices** dropdown (set to Mono)
- Buffer Size** dropdown (set to 1024)
- Use AudioWorklet** checkbox (unchecked)
- Save DSP Code** checkbox (checked)
- Save Params State** checkbox (unchecked)
- Save DSP Cache** checkbox (unchecked)
- Real-time Compile** checkbox (checked)
- PopUp UI** checkbox (unchecked)
- Plot** section:
  - Mode** dropdown (set to Offline)
  - Samples** dropdown (set to 1024)
  - Sample Rate** dropdown (set to 48000)
  - FFT Size** dropdown (set to 256)
  - Draw Spectrogram** checkbox (unchecked)
  - Plot First Samples** button

**Center Panel (Code Editor):**

The code editor shows the DSP code for a blipper effect. The code is written in Faust and includes comments in Arabic. The code defines parameters for BasePitch, PitchMod, Attack, Release, and Mix, and implements the blipper function.

```
13 block_on(fx) = par(1, inputs(fx), _<{bypass});
14 block_off(fx) = par(1, inputs(fx), _bypass);
15
16 bypass_fx(fx) = par(1, inputs(fx), _<{(block_on(fx):fx), block_off(fx)}> par(1, outputs(fx), _);
17
18 basepitch = hslider("BasePitch [unit:semitones] [Owl:PARAMETER_A] [style:knob]", 60, 24, 96, 0.1) : si.smooth(ba.tau2pole(0.01));
19 pitchmod = hslider("PitchMod [unit:semitones] [Owl:PARAMETER_B] [style:knob]", 24, -64, 64, 1) : si.smooth(ba.tau2pole(0.005));
20 //attack = hslider("Attack [unit:ms] [Owl:PARAMETER_C] 2, 2, 1000, 1) : *(0.001) : max(1.0/float(ma.SR));
21 release = hslider("Release [unit:ms] [Owl:PARAMETER_C] [style:knob]", 20, 2, 100, 1) : *(0.001) : max(1.0/float(ma.SR));
22 attack = 0.005;
23 mix = hslider("Mix [Owl:PARAMETER_D] [style:knob]", 0.5, 0, 1, 0.01) : si.smooth(ba.tau2pole(0.005));
24
25 blipper(l, r) = l, r <: *(1-mix), *(1-mix), mono2stereo := _;
26
27 with {
28   mono2stereo = + : pc2 * mix <: _;
29   pc2 = an.amp_follower_ud(attack, release) <: (ba.midikey2hz(basepitch + (pitchmod * _)) : os.triangle, _ : *);
30 };
31 process = bypass_fx(blipper);
```

**Right Panel (GUI Builder):**

The GUI Builder section shows the visual representation of the DSP code. It includes a "Welcome on the WAP GUI2 editor ! click on MENU to start !" message and a preview of the GUI. The GUI features four knobs labeled BasePitch, Mix, PitchMod, and bypassRelease, and a background image of a brick wall.

# Online testers (individual plugin, repository)

← → ↺ 🏠 **Sécurisé** | <https://wasabi.i3s.unice.fr/plugins/delay-plugin>

Applications 🔍 AUFX-O: The Audio... LV2 programming fo... Bundle Defi

## Plugin Tester

▶ 0:00 / 2:56 🔊 ⬇

**Paste here the link to your webaudio plugin**

URL: <https://wasabi.i3s.unice.fr/>



100% passes: 12 failures: 5 duration: 0.05s

### Metadata

- ✓ plugin should have a `JSON getMetadata()` method
- ✓ the `getMetadata()` function should return a json object

### Descriptor

- ✓ plugin should have a `JSON getDescriptor()` method
- ✓ `getDescriptor()` function should return a json object

### Param getter

- ✗ plugin should have a `getParam(key)` method

```
AssertionError: expected undefined to exist
at Context.<anonymous> (test.html:135:54)
```

- ✗ the `getParam()` function should not be empty

```
AssertionError: .empty was passed non-string primitive undefined
at Context.<anonymous> (test.html:138:61)
```

← → ↺ 🏠 **Sécurisé** | <https://wasabi.i3s.unice.fr/WebAudioPluginBank/testers/explorandtest.html>

Applications 🔍 HTML5's Media Re... faust/examples at... JS Bin - Collaborati... surikow/zyvoog: Plug... Microsoft Word - Q... (63) Greta Van Flee...

▶ 0:00 / 2:56 🔊 ⬇

<https://wasabi.i3s.unice.fr/WebAudioPluginBank/repository.json>



**WasabiQuadrFuzz**

100% passes: 14 failures: 0 duration: 0.05s

### Metadata

- ✓ plugin should have a `JSON getMetadata()` function

### Descriptor

- ✓ plugin should have a `JSON getDescriptor()` function

### Param getter

- ✓ plugin should have a `getParam(key)` method

### Param setter

- ✓ plugin should have a `setParam(key, value)` method

← → ↺ 🏠 **Sécurisé** | <https://wasabi.i3s.unice.fr/WebAudioPluginBank/testers/S...>

Applications 🔍 Making it easier to... The online IDE for ... 2017 JavaScript RL... The POLICE Walk... Pink Floyd Backlin...

▶ 0:00 / 2:56 🔊 ⬇

## Plugin Tester

**-Paste here the link to your webaudio plugin**

URL: <https://wasabi.i3s.unice.fr/WebAudioPluginBank/WASABI/PingPongDelay3/>

Click to fill with the URL of :

- WasabiPingPongDelay - FaustZitaRev - LarkinBlipper - LarkinFlanger
- WasabiMinilogue - CleanMachine - QuadraFuzz

**WasabiPingPongDelay**



**Node Part**

URL: node.URL

"<https://wasabi.i3s.unice.fr/WebAudioPluginBank/WASABI/PingPongDelay3/>"

Descriptor : `node.getDescriptor()`

```
{
  "mix": {
    "minValue": 0,
    "maxValue": 1,
    "defaultValue": 0.5,
    "time": {
      "minValue": 0,
      "maxValue": 1,
      "defaultValue": 0.5,
      "feedback": {
        "minValue": 0,
        "maxValue": 1,
        "defaultValue": 0.5
      }
    }
  }
}
```

Node input number `node.numberOfInputs : 1`

Params at this time `node.getState()` (promise)

```
{
  "mix": 0.5,
  "time": 0.5,
  "feedback": 0.5,
  "status": "disable"
}
```

Metadata `node.getMetadata()` (promise)

```
{
  "documentation": "...",
  "name": "PingPongDelay",
  "thumbnail": "WasabiPing..."
}
```

# Conclusion : Where are we today?

SDK for JS developers

FAUST scripts and IDE that compile .dsp files to WAPs, embedded GUIs builder, publish to remote servers

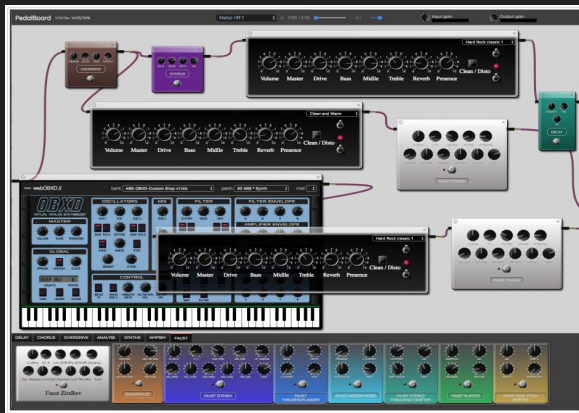
WebAudioModules C/C++ toolchain for native audio developers (and WAMs are WAPs)

Multiple examples of hosts that load plugins dynamically using URIs

Tools: plugin validator, repository validator, GUI editor

[Check the GitHub!](#)

[Check the pedalboard demo!](#)





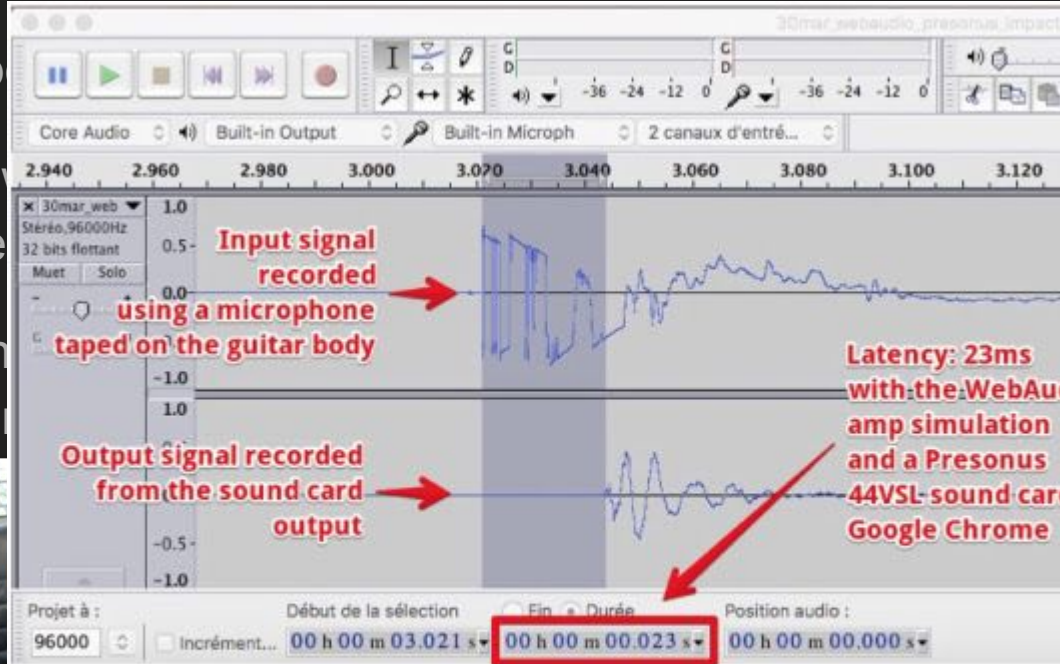


# Guitar Hero / Rocksmith for the Web?

Yes, this is possible

- Good low latency
- Pitch detection

Still the latency is addressed on the Web



**Figure 7: Example of measure in Audacity, here the WebAudio amp sim with Google Chrome and a Presonus 44VSL sound card.**

