

MIDI IS STAGING
A COMEBACK

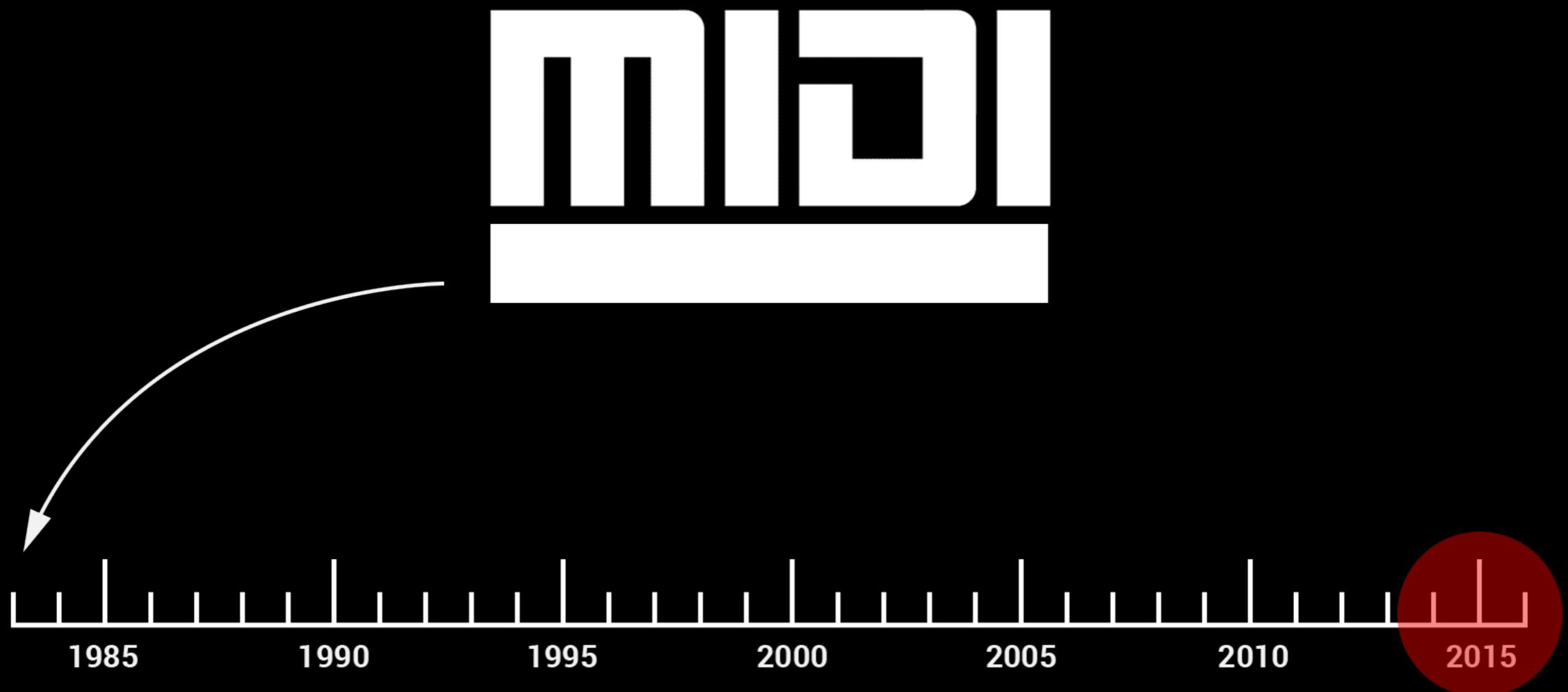
In Your Browser!

GREETINGS PROFESSOR FALKEN.
SHALL WE PLAY?

CHESS
POKER
FIGHTER COMBAT
AIR-TO-GROUND ACTIONS
>YOU GIVE LOVE A BAD NAME. MID
GLOBAL THERMONUCLEAR WAR

THE ONLY WINNING MOVE IS
NOT TO PLAY.





WEB MIDI API

*« the Web-MIDI API is
the most significant advancement
of MIDI since... MIDI itself! »*

– [midi.org](https://www.midi.org/articles/about-web-midi)

Web MIDI API

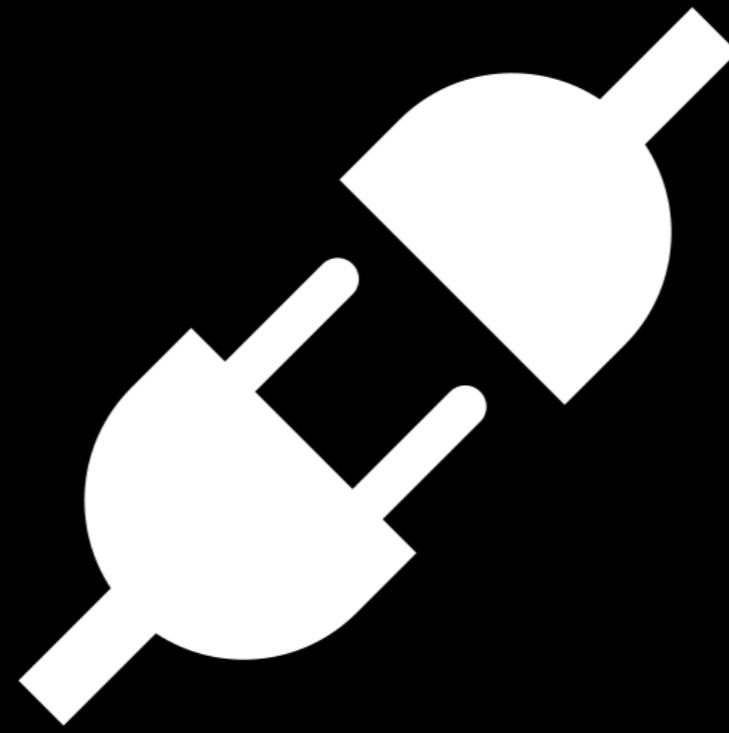
The Web MIDI API specification defines a means for web developers to enumerate, manipulate and access MIDI devices

Global	53.77%
Canada	48.12%

IE	Edge *	Firefox	Chrome	Safari	Opera	iOS Safari *	Opera Mini *	Android Browser	Chrome for Android
			49					4.4	
8	13	47	51			9.2		4.4.4	
11	14	48	52	9.1	39	9.3	all	51	51
		49	53	10	40				
		50	54	TP	41				
		51	55						



Usually based on Chromium
(the open source, seed version of Chrome)



jazz-soft.net



Not
Currently
Planned



Development
Started



?

SUPPORTED

87%

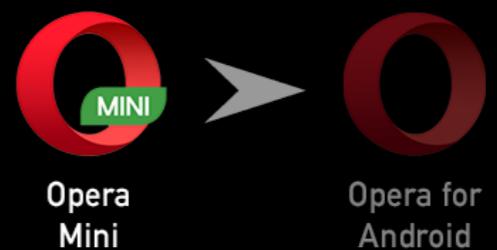
natively or with plugin

NO SUPPORT

4.88%

8.16%

0.16%



Opera
Mini



Safari
on iOS



Edge



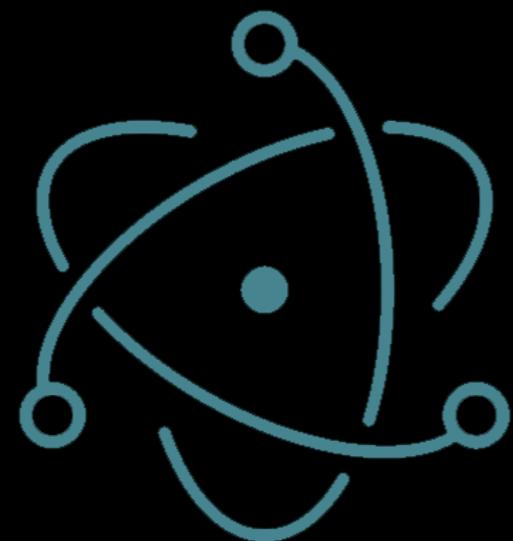
Opera
for
Android



Web MIDI
Browser



Internet
Explorer

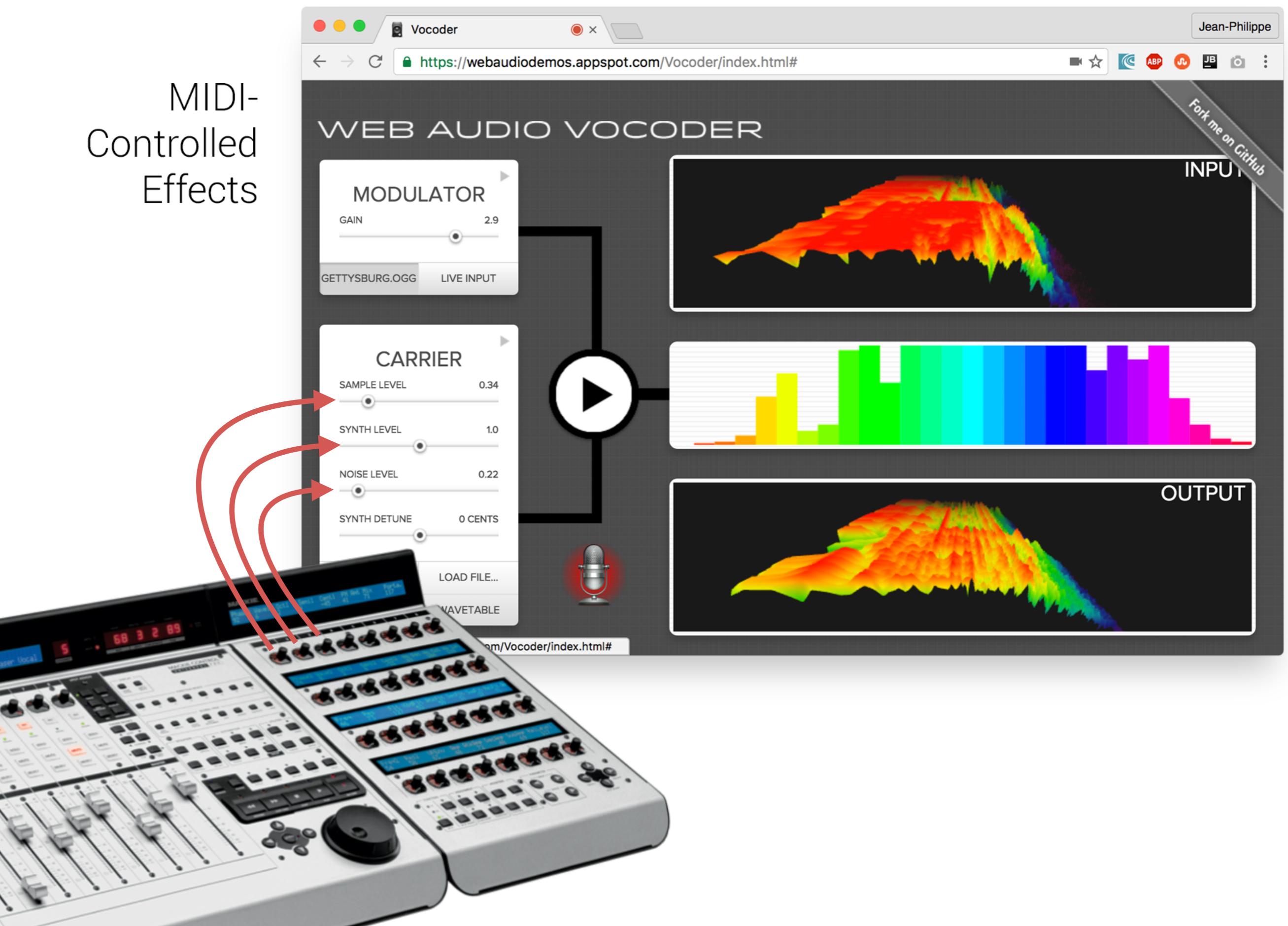


Electron
electron.atom.io



NW.js
nwjs.io

MIDI- Controlled Effects



Editors & Librarians



This screenshot shows the JP-08 Editor software interface. The top half contains six horizontal panels for LFO, VCO Mod, VCO-1, VCO-2, and VCO-1 VCO-2. Each panel has various knobs and dropdown menus for waveform selection and modulation depth. Below these are three more panels: HPF, VCF, and VCA, followed by ENV-1 and ENV-2. The bottom section includes buttons for Delay, Portamento, Assign Mode, Bend Range, Patch, and Set Up, along with a Connexion tab and a My patches dropdown.

This screenshot shows the JP-08 Editor software interface. It includes a central Arpeggiator setup section with fields for JP08 MIDI In, JP08 MIDI Out, and Keyboard MIDI In, along with buttons for Update Connections and Resend Current Patch. To the right is a large button labeled 'JP-08 Editor' and 'Donate'.

This screenshot shows the JX-03 Editor software interface. It features a grid of knobs and buttons for DCO-2, VCF, VCA, LFO, and Envelope sections. Below this is a patch selection section with buttons for Chorus, Delay, Portamento, Assign Mode, Bend Range, Patch, and Set Up. A Connexion tab and a My patches dropdown are also present.

The image displays three separate browser windows showcasing different web-based audio applications:

- Shiny Happy WebAudio MIDI-fx**: A WebAudio Drum Machine 1.0 interface. It features a grid of 24 circular buttons representing different drum sounds (Tom 1, Tom 2, Tom 3, Hi-Hat, Snare, Kick). Below the grid are controls for Kit (Kit 3), Effect (Living Room), Tempo (100 bpm), and Swing. There are also sliders for Effect Level, Kick Pitch, and Snare Pitch. At the bottom are buttons for Beat (play/pause), Save, Load, Reset, and Demo. Input and output ports are set to Axiom Pro 25 and Gestionnaire IAC Bus 1 respectively.
- Web Audio MIDI Synthesizer**: A complex analog-style synthesizer interface. It includes two oscillators (OSC1 and OSC2) with waveform selection (sawtooth, sine, etc.), frequency modulation (tremolo), and filter controls (cutoff, resonance). It features two filter envelopes (filter envelope 1 and filter envelope 2) with attack, decay, sustain, and release parameters. A master section includes Drive, Reverb, and Volume controls. A piano-roll style keyboard is at the bottom.
- WebSynths**: An oscillator and filter configuration interface. It allows setting BPM (beats per minute), oscillator parameters (shape, waveform, super size, frequency modifiers, amplitude modifiers), and filter settings (filters, filter matrix, to filter 1, to filter 2, to filters out, from filter 1, to filter 2). It also includes modulators like chorus and stereo invert. At the bottom, there are controls for pitch system (equal tempered), number of pads (15), key (none), and start note (4 - C).

Synths & Drum Machines

WebSynths by Mitch Wells

The screenshot shows the WebSynths interface, a web-based digital audio workstation. The top navigation bar includes tabs for 'load', 'save', 'shop', and social media links. The main interface is divided into three main sections: 'bpm' (beats per minute), 'oscillator 1' (with shape, waveform, pulse width, super size, frequency modifiers, and shift controls), and 'filters' (with filter matrix, filter 1 type set to lowpass, cutoff (Hz) at 350, frequency follow, cutoff lfo, cutoff envelope, emphasis, and filter 2). The 'effects' section includes compressor, distortion, amount (set to 11), eq, level (%), mix (%), modulators, and delays. A 'tuning' section at the bottom provides a grid of notes from C (523 Hz) to A (1760 Hz), with specific frequencies listed below each note.

z	x	c	v	b	n	m	<	>	?	a	s	d	f	g	h	j	k	l	:	q	w
C 523 Hz	C# 554 Hz	D 587 Hz	D# 622 Hz	E 659 Hz	F 698 Hz	F# 740 Hz	G 784 Hz	G# 831 Hz	A 880 Hz	A# 932 Hz	B 988 Hz	C 1047 Hz	C# 1109 Hz	D 1175 Hz	D# 1245 Hz	E 1319 Hz	F 1397 Hz	F# 1480 Hz	G 1568 Hz	G# 1661 Hz	A 1760

Web MIDI API Demo

index.html

```
<!DOCTYPE html>
<html lang="en">
  <head>
    <meta charset="UTF-8">
    <title>Web MIDI API Demo</title>
    <!-- Web MIDI API Polyfill -->
    <script src="WebMIDIAPIShim.min.js"></script>
    <!-- Our code -->
    <script src="script.js"></script>
  </head>
  <body>
    <h1>Web MIDI API Demo</h1>
    <p>Open the console to view results.</p>
  </body>
</html>
```

Web MIDI API Demo

script.js v1

```
addEventListener("DOMContentLoaded", function () {  
  
  if (!navigator.requestMIDIAccess) {  
    console.log("MIDI is not supported by your browser.");  
    return;  
  }  
  
  navigator.requestMIDIAccess().then(onSuccess, onFailure);  
  
  function onFailure(e) {  
    console.log("MIDI cannot be activated.\n\n" + e);  
  }  
  
  function onSuccess(midiAccess) {  
    console.log(midiAccess);  
  }  
});
```



```
▼ MIDIAccess  
► inputs: MIDIIInputMap  
onstatechange: null  
► outputs: MIDIOutputMap  
sysexEnabled: false  
► __proto__: MIDIAccess
```

Web MIDI API Demo

script.js v2

```
▶ Object
▶ Object
▼ Object
  done: false
  ▼ value: MIDIOutput
    connection: "open"
    id: "1584982307"
    manufacturer: "M-Audio"
    name: "M-Audio Accent"
    onstatechange: null
    state: "connected"
    type: "output"
    version: ""
  }
```

```
addEventListener("DOMContentLoaded", function () {

  if (!navigator.requestMIDIAccess) {
    console.log("MIDI is not supported by your browser.");
    return;
  }

  navigator.requestMIDIAccess().then(onSuccess, onFailure);

  function onFailure(e) {
    console.log("MIDI cannot be activated.\n\n" + e);
  }

  function onSuccess(midiAccess) {

    var outputs = midiAccess.outputs.values();

    for (var output = outputs.next(); output && !output.done;
         output = outputs.next()) {

      ← console.log(output);

      if (output.value.name === "M-Audio Accent") {
        var synth = output.value;
      }
    }
  }
});
```

Web MIDI API Demo

script.js v3

```
addEventListener("DOMContentLoaded", function () {

  if (!navigator.requestMIDIAccess) {
    console.log("MIDI is not supported by your browser.");
    return;
  }

  navigator.requestMIDIAccess().then(onSuccess, onFailure);

  function onFailure(e) {
    console.log("MIDI cannot be activated.\n\n" + e);
  }

  function onSuccess(midiAccess) {

    var outputs = midiAccess.outputs.values();

    for (var output = outputs.next(); output && !output.done;
        output = outputs.next()) {

      if (output.value.name === "M-Audio Accent") {
        var synth = output.value;
      }
    }

    synth.send(???);
  }
});
```

EXPANDED MESSAGES LIST (STATUS BYTES)

STATUS BYTE		DATA BYTES	
1st Byte Value	Function	2nd Byte	3rd Byte
10000000= 80= 128	Chan 1 Note off	Note Number (0-127)	Note Velocity (0-127)
10000001= 81= 129	Chan 2 Note off	Note Number (0-127)	Note Velocity (0-127)
10000010= 82= 130	Chan 3 Note off	Note Number (0-127)	Note Velocity (0-127)
10000011= 83= 131	Chan 4 Note off	Note Number (0-127)	Note Velocity (0-127)
10000100= 84= 132	Chan 5 Note off	Note Number (0-127)	Note Velocity (0-127)
10000101= 85= 133	Chan 6 Note off	Note Number (0-127)	Note Velocity (0-127)
10000110= 86= 134	Chan 7 Note off	Note Number (0-127)	Note Velocity (0-127)
10000111= 87= 135	Chan 8 Note off	Note Number (0-127)	Note Velocity (0-127)
10001000= 88= 136	Chan 9 Note off	Note Number (0-127)	Note Velocity (0-127)
10001001= 89= 137	Chan 10 Note off	Note Number (0-127)	Note Velocity (0-127)
10001010= 8A= 138	Chan 11 Note off	Note Number (0-127)	Note Velocity (0-127)
10001011= 8B= 139	Chan 12 Note off	Note Number (0-127)	Note Velocity (0-127)
10001100= 8C= 140	Chan 13 Note off	Note Number (0-127)	Note Velocity (0-127)
10001101= 8D= 141	Chan 14 Note off	Note Number (0-127)	Note Velocity (0-127)
10001110= 8E= 142	Chan 15 Note off	Note Number (0-127)	Note Velocity (0-127)
10001111= 8F= 143	Chan 16 Note off	Note Number (0-127)	Note Velocity (0-127)
10010000= 90= 144	Chan 1 Note on	Note Number (0-127)	Note Velocity (0-127)
10010001= 91= 145	Chan 2 Note on	Note Number (0-127)	Note Velocity (0-127)

Web MIDI API Demo

script.js v4

```
addEventListener("DOMContentLoaded", function () {

  if (!navigator.requestMIDIAccess) {
    console.log("MIDI is not supported by your browser.");
    return;
  }

  navigator.requestMIDIAccess().then(onSuccess, onFailure);

  function onFailure(e) {
    console.log("MIDI cannot be activated.\n\n" + e);
  }

  function onSuccess(midiAccess) {

    var outputs = midiAccess.outputs.values();

    for (var output = outputs.next(); output && !output.done;
        output = outputs.next()) {

      if (output.value.name === "M-Audio Accent") {
        var synth = output.value;
      }
    }

    synth.send([???], 48, 127);
  }
});
```

Octave	Note Numbers											
	C	C#	D	D#	E	F	F#	G	G#	A	A#	B
-2	0	1	2	3	4	5	6	7	8	9	10	11
-1	12	13	14	15	16	17	18	19	20	21	22	23
0	24	25	26	27	28	29	30	31	32	33	34	35
1	36	37	38	39	40	41	42	43	44	45	46	47
2	48	49	50	51	52	53	54	55	56	57	58	59
3	60	61	62	63	64	65	66	67	68	69	70	71
4	72	73	74	75	76	77	78	79	80	81	82	83
5	84	85	86	87	88	89	90	91	92	93	94	95
6	96	97	98	99	100	101	102	103	104	105	106	107
7	108	109	110	111	112	113	114	115	116	117	118	119
8	120	121	122	123	124	125	126	127				

Web MIDI API Demo

script.js v5

```
if (!navigator.requestMIDIAccess) {
    console.log("MIDI is not supported by your browser.");
    return;
}

navigator.requestMIDIAccess().then(onSuccess, onFailure);

function onFailure(e) {
    console.log("MIDI cannot be activated.\n\n" + e);
}

function onSuccess(midiAccess) {

    var outputs = midiAccess.outputs.values();

    for (var output = outputs.next(); output && !output.done;
        output = outputs.next()) {

        if (output.value.name === "M-Audio Accent") {
            var synth = output.value;
        }
    }

    var command = 0x9;
    var channel = 1;

    var status = (command << 4) + (channel - 1);
    synth.send([status, 48, 127]);
}

});
```

#%?*&\$#!

WEBMIDI LIBRARY

<https://github.com/cotejp/webmidi>

webmidi library demo

index.html

```
<!DOCTYPE html>

<html lang="en">

  <head>

    <meta charset="UTF-8">
    <title>Demo Using the 'webmidi' Library</title>

    <!-- Web MIDI API Polyfill -->
    <script src="WebMIDIAPI.min.js"></script>

    <!-- webmidi Library -->
    <script src="webmidi.min.js"></script>

    <!-- Our code -->
    <script src="script.js"></script>

  </head>

  <body>
    <h1>
      Demo Using the 'webmidi' Library (available on
      <a href="https://github.com/cotejp/webmidi">GitHub</a>).
    </h1>
    <p>Open the console to view results.</p>
  </body>

</html>
```

webmidi library demo

script.js v1

```
addEventListener("DOMContentLoaded", function () {
  WebMidi.enable(function (err) {
    if (err) {
      console.log("WebMidi could not be enabled.", err);
      return;
    }
    console.log(WebMidi.outputs);
  });
});
```

▼ Array[3] ⓘ
▶ 0: Output
▶ 1: Output
▼ 2: Output
▶ _midiOutput: MIDIOutput
connection: "closed"
▶ get connection: *function ()*
id: "1584982307"
▶ get id: *function ()*
manufacturer: (...)
▶ get manufacturer: *function ()*
name: "M-Audio Accent"
▶ get name: *function ()*
state: "connected"
▶ get state: *function ()*

webmidi library demo

script.js v2

```
addEventListener("DOMContentLoaded", function () {  
  
  WebMidi.enable(function (err) {  
  
    if (err) {  
      console.log("WebMidi could not be enabled.", err);  
      return;  
    }  
  
    WebMidi  
      .getOutputByName("M-Audio Accent")  
      .playNote("C2");  
  
  });  
});
```

Functions for outgoing MIDI messages

- decrementRegisteredParameter
- incrementRegisteredParameter
- playNote
- send
- sendActiveSensing
- sendChannelAftertouch
- sendChannelMode
- sendClock
- sendContinue
- sendControlChange
- sendKeyAftertouch
- sendPitchBend
- sendProgramChange
- sendReset
- sendSongPosition
- sendSongSelect
- sendStart
- sendStop
- sendSysex
- sendTimecodeQuarterFrame
- sendTuningRequest
- setMasterTuning
- setModulationRange
- setNonRegisteredParameter
- setPitchBendRange
- setRegisteredParameter
- setTuningBank
- setTuningProgram
- stopNote

Events for incoming MIDI messages

- activesensing
- channelaftertouch
- channelmode
- clock
- continue
- controlchange
- keyaftertouch
- noteoff
- noteon
- pitchbend
- programchange
- reset
- songposition
- songselect
- start
- stop
- sysex
- timecode
- tuningrequest

WEBMIDI LIBRARY



<https://github.com/cotejp/webmidi>

**THIS SIDE
INTENTIONALLY
LEFT BLACK**

USED Yamaha KX-5 w/Hard Case

Item **Used**
condition:

"This does not have a lid for the battery box but Unit is in perfect working order. Please see photos"

[... Read more](#)

Sold for: **US \$349.99**

Approximately

C \$462.55

[Add to list](#) ▾

Shipping: **US \$130.00 (approx. C \$171.81)** Expedited Int'l Shipping | [See details](#)

International items may be subject to customs processing and additional charges. [?](#)

Item location: Yokohama, Japan

Ships to: Worldwide

Delivery: Estimated Delivery within 13-18 business days

Seller ships within 10 days after [receiving cleared payment](#).

[?](#)

Please allow additional time if international delivery is subject to customs processing.

Payments: [PayPal](#) [VISA](#) [MasterCard](#) [Discover](#) [American Express](#)

Credit cards processed by PayPal

[See payment information](#)

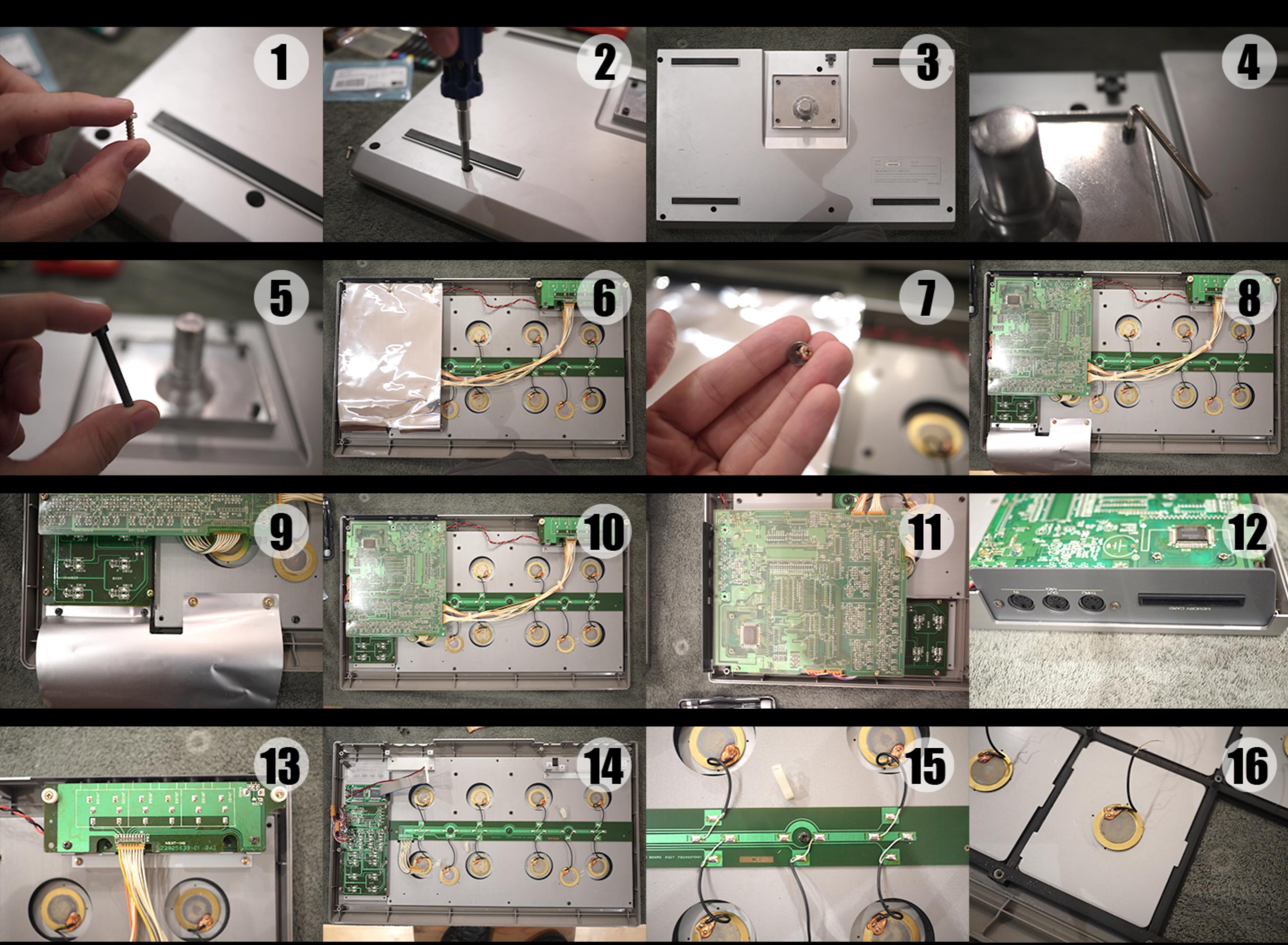
Returns: 14 day money back, buyer pays return shipping | [See details](#)



Roland Octapad II (PAD-80)



1988









CODE DEMO



OCTAPAD

MIDI CABLE

USB CABLE

COMPUTER

The screenshot shows a software interface for developing web applications. On the left, the **Project** panel displays the file structure of the project "03-octapad-demo". A red box highlights the following files and folders:

- images**: Contains `octapad.svg`.
- libs**: Contains `snap.svg-min.js`, `Tone.min.js`, and `webmidi.min.js`.
- samples**: Contains audio files: `bass.wav`, `bongohi.wav`, `bongolo.wav`, `hat.wav`, `hit.wav`, `kick.wav`, `shaker.wav`, `snare.wav`, and `tambourine.wav`.
- index.html**: The main HTML file.
- script.js**: An external JavaScript file.

The **index.html** file is open in the central editor window. The code is as follows:

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>Octapad Demo!</title>
    <!-- Libraries -->
    <script src="libs/webmidi.min.js"></script>
    <script src="libs/Tone.min.js"></script>
    <script src="libs/snap.svg-min.js"></script>
    <style>
        html { background-color: black; }
    </style>
    <!-- Our Code-->
    <script src="script.js"></script>
</head>
<body></body>
</html>
```

The **Structure** panel on the left shows the current file structure. The **Remote Host** panel on the right indicates that the file is successfully connected to a remote host.

```
document.addEventListener("DOMContentLoaded", function () {  
});
```

```
document.addEventListener("DOMContentLoaded", function () {  
    var snap, octapad, input, player;  
});
```

```
document.addEventListener("DOMContentLoaded", function () {  
    var snap, octapad, input, player;  
  
    snap = Snap(window.innerWidth, window.innerHeight);  
});
```

```
document.addEventListener("DOMContentLoaded", function () {  
    var snap, octapad, input, player;  
  
    snap = Snap(window.innerWidth, window.innerHeight);  
    octapad = snap.group();  
  
});
```

```
document.addEventListener("DOMContentLoaded", function () {  
  var snap, octapad, input, player;  
  
  snap = Snap(window.innerWidth, window.innerHeight);  
  octapad = snap.group();  
  
  Snap.load("images/octapad.svg", function (image) {  
    octapad.append(image);  
    octapad.animate({transform: "s.9,.9"}, 1000);  
  });  
});
```

```
document.addEventListener("DOMContentLoaded", function () {  
  var snap, octapad, input, player;  
  
  snap = Snap(window.innerWidth, window.innerHeight);  
  octapad = snap.group();  
  
  Snap.load("images/octapad.svg", function (image) {  
    octapad.append(image);  
    octapad.animate({transform: "s.9,.9"}, 1000);  
  });  
  
  WebMidi.enable(function(err) {  
    if (err) { throw new Error("WebMidi could not be enabled."); }  
  });  
});
```

```
document.addEventListener("DOMContentLoaded", function () {  
  var snap, octapad, input, player;  
  
  snap = Snap(window.innerWidth, window.innerHeight);  
  octapad = snap.group();  
  
  Snap.load("images/octapad.svg", function (image) {  
    octapad.append(image);  
    octapad.animate({transform: "s.9,.9"}, 1000);  
  });  
  
  WebMidi.enable(function(err) {  
    if (err) { throw new Error("WebMidi could not be enabled."); }  
  
    input = WebMidi.getInputByName("MIDISPORT 2x2 Port A");  
    setUpSamplePlayer();  
  });  
});
```

```
document.addEventListener("DOMContentLoaded", function () {  
  var snap, octapad, input, player;  
  
  snap = Snap(window.innerWidth, window.innerHeight);  
  octapad = snap.group();  
  
  Snap.load("images/octapad.svg", function (image) {  
    octapad.append(image);  
    octapad.animate({transform: "s.9,.9"}, 1000);  
  });  
  
  WebMidi.enable(function(err) {  
    if (err) { throw new Error("WebMidi could not be enabled."); }  
  
    input = WebMidi.getInputByName("MIDISPORT 2x2 Port A");  
    setUpSamplePlayer();  
  });  
  
  function setUpSamplePlayer() {  
  
    player = new Tone.MultiPlayer(sampleMap, addInputListener).toMaster();  
  }  
});
```

```
WebMidi.enable(function(err) {
  if (err) { throw new Error("WebMidi could not be enabled."); }

  input = WebMidi.getInputByName("MIDISPORT 2x2 Port A");
  setUpSamplePlayer();
});

function setUpSamplePlayer() {

  var sampleMap = {
    48: "samples/tambourine.wav",    // PAD1
    45: "samples/bongolo.wav",      // PAD2
    41: "samples/bongohi.wav",      // PAD3
    51: "samples/shaker.wav",       // PAD4
    35: "samples/snare.wav",        // PAD5
    38: "samples/bass.wav",         // PAD6
    42: "samples/hit.wav",          // PAD7
    49: "samples/hat.wav"           // PAD8
  };

  player = new Tone.MultiPlayer(sampleMap, addInputListener).toMaster();
}

});
```

```
WebMidi.enable(function(err) {
  if (err) { throw new Error("WebMidi could not be enabled."); }

  input = WebMidi.getInputByName("MIDISPORT 2x2 Port A");
  setUpSamplePlayer();
});

function setUpSamplePlayer() {

  var sampleMap = {
    48: "samples/tambourine.wav",    // PAD1
    45: "samples/bongolo.wav",      // PAD2
    41: "samples/bongohi.wav",      // PAD3
    51: "samples/shaker.wav",       // PAD4
    35: "samples/snare.wav",        // PAD5
    38: "samples/bass.wav",         // PAD6
    42: "samples/hit.wav",          // PAD7
    49: "samples/hat.wav"           // PAD8
  };

  player = new Tone.MultiPlayer(sampleMap, addInputListener).toMaster();
}

function addInputListener() {

  input.addListener('noteon', "all", function(e) {

  });
}

});
```

```
function setUpSamplePlayer() {  
  
    var sampleMap = {  
        48: "samples/tambourine.wav", // PAD1  
        45: "samples/bongolo.wav", // PAD2  
        41: "samples/bongohi.wav", // PAD3  
        51: "samples/shaker.wav", // PAD4  
        35: "samples/snare.wav", // PAD5  
        38: "samples/bass.wav", // PAD6  
        42: "samples/hit.wav", // PAD7  
        49: "samples/hat.wav" // PAD8  
    };  
  
    player = new Tone.MultiPlayer(sampleMap, addInputListener).toMaster();  
}  
  
function addInputListener() {  
  
    input.addListener('noteon', "all", function(e) {  
  
        // Play sound (buffer to play, delay, offset inside buffer, duration, pitch shift, gain)  
        player.start(e.note.number, undefined, 0, undefined, 0, e.velocity);  
    });  
}  
});
```

```
function setUpSamplePlayer() {
    var sampleMap = {
        48: "samples/tambourine.wav", // PAD1
        45: "samples/bongolo.wav", // PAD2
        41: "samples/bongohi.wav", // PAD3
        51: "samples/shaker.wav", // PAD4
        35: "samples/snare.wav", // PAD5
        38: "samples/bass.wav", // PAD6
        42: "samples/hit.wav", // PAD7
        49: "samples/hat.wav" // PAD8
    };

    player = new Tone.MultiPlayer(sampleMap, addInputListener).toMaster();
}

function addInputListener() {
    input.addListener('noteon', 'all', function(e) {
        // Play sound (buffer to play, delay, offset inside buffer, duration, pitch shift, gain)
        player.start(e.note.number, undefined, 0, undefined, 0, e.velocity);

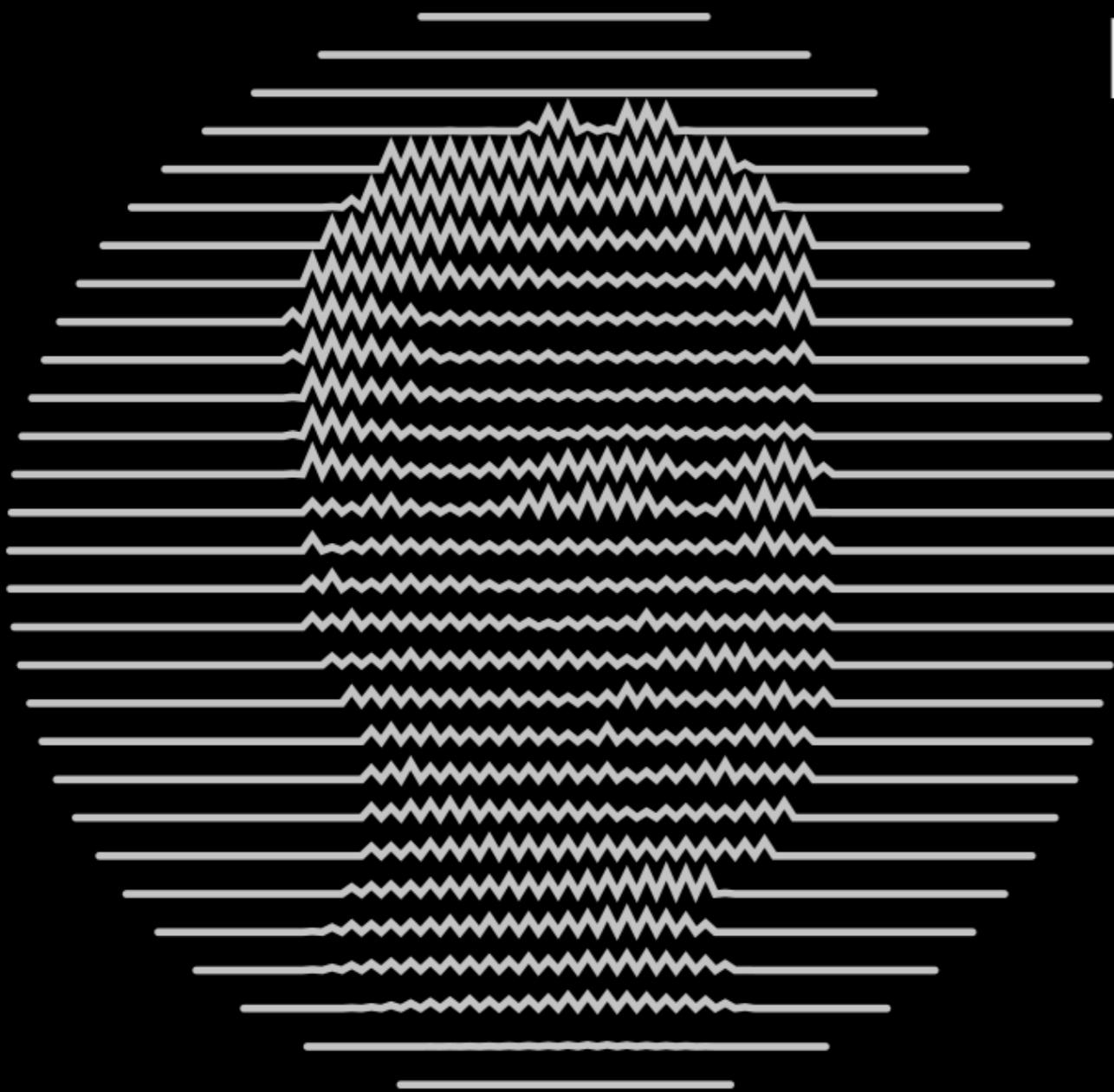
        var map = { 48: 1, 45: 2, 41: 3, 51: 4, 35: 5, 38: 6, 42: 7, 49: 8 };
        octapad.select("#pad" + map[e.note.number])
            .attr("fill", "white")
            .animate({fill: "black"}, 500, mina.easeout);
    });
}

});
```

LET'S HEAR IT OUT

HOMEWORK





<http://tangiblejs.com/webu2016>



<http://tangiblejs.com>



@tangiblejs



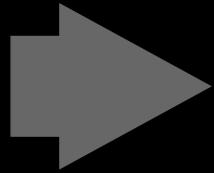
<http://cote.cc>



@jcote

<http://github.com/cotejp>

MERCI.



Vote for Web MIDI API Support in Microsoft Edge

<https://wpdev.uservoice.com/forums/257854-microsoft-edge-developer/suggestions/6508429-web-midi-api>

Follow Web MIDI Development in Firefox

https://bugzilla.mozilla.org/show_bug.cgi?id=836897

Web MIDI API Specification

<https://webaudio.github.io/web-midi-api/>

WebMidi Library

<https://github.com/cotejp/webmidi>

Web MIDI API Polyfill

<http://cwilso.github.io/WebMIDIAPIShim/>

Web MIDI API Shim for iOS

<https://github.com/mizuhiki/WebMIDIAPIShimForiOS>

Web MIDI Browser

<http://www.taktech.org/takm/WebMIDIBrowser/>

Tone.js Web Audio Framework

<https://github.com/Tonejs/Tone.js/>

Code From This Talk

<https://github.com/cotejp/web-unleashed-2016-midi-talk>