Audio on the Web (a brief history)
Michel Buffa

To cite this version:
Michel Buffa. Audio on the Web (a brief history). SMC 2022 - Sound and Music Computing, Jun 2022, Saint-Etienne, France. hal-03871527

HAL Id: hal-03871527
https://inria.hal.science/hal-03871527
Submitted on 25 Nov 2022

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.
Audio on the Web (a brief history)

Sound and Music Computing 2022
11 June 2022

Michel Buffa,
Université Côte d'Azur, France,
I3S/CNRS/INRIA labs
buffa@i3s.unice.fr, @micbuffa
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
  - W3C Advisory Committee Representative for UCA

- I participate to W3C working groups (HTML5, WebAudio…)
- I’m the co-creator of the Web Audio Modules 2.0 (WAM or WAM2), aka VSTs for the Web.
- Some WebAudio plugins I wrote are now commercialized by CNRS (tube guitar amplifier simulators + audio FXs)
Before 1996 we barely could hear any sound from a browser.
Welcome to Michel's home page! 
buffa@unice.fr

Who am I? What is my work...
My coordinates
My hobbies
My courses

You want to see me live? Check this funny anim and also this new one!

Table of contents

- Internet, WWW, etc...
- Teaching computer techniques, University of Nice related stuff
- Research centers I have worked with
- Culture...
- Unix related stuff, X11/Motif/Tk, ftp, fsp, etc...
- Virtual Reality/3D/Computer Graphics
- Dictionaries, maps, etc...
- Interesting personal Home Pages
- Pure Fun
- Business WWW sites

My courses

- Computer Graphics and OpenGL Programming
- Internet and WWW
- Java
- X11 and Motif Programming
- VRML
- Zsh
- An old WWW tutorial I wrote in 1994
  others...

Created in 1993, last updated: 1996, HTML 1
After 1996
Horrors and Joys
The infamous `<bgsound>` tag from IE4! (1996)

...also known as the worst possible way to use audio on the web!

**Supported Browsers:** The supported browsers by `HTML | <bgsound> tag` are listed below:

- Internet Explorer

I assume you are using the bgsound tag.

> The bgsound tag is non-standard and IE-specific. Firefox does not support it.

I would strongly suggest that you do not use a background sound, unless this is a personal/family site where your friends and family know what to expect.

As a rule, visitors do not appreciate their family, co-workers, other coffee shop patrons, or sleeping babies being blasted with sound from their computer that they had no control over.

If you want to play sound, do it with a method that lets the visitor control starting it and stopping it; don't try to do it automatically.
Error loading media: File could not be played
However, with a little struggle, we had great applications!

FXExperiencePlayer / Java FX (video from 2012)

Audiotools / Flash (video from 2013)
2008-2010: HTML5 audio tag with rather complete JavaScript API.

ANOTHER CODEC WAR, until 2015 no way to play a mp3 on all major browsers!
For serious audio applications: Flash or Die!

Soundation

Audio Sauna
2010
First browser APIs
for audio processing
Firefox Audio Data API (2010)

Simple: nearly no native code in the browser, all sound processing delegated to JavaScript, short spec!

A number of working demos have been created, including:

**NOTE:** If you try to run demos created with the original API using a build that implements the new API, you may encounter bug 560212. We are aware of this, as is Mozilla, and it is being investigated.

**Demos Working on Current API**

- FFT visualization (calculated with js)
  - [http://ware.buildingsky.net/processing/dsp.js/examples/fft.html](http://ware.buildingsky.net/processing/dsp.js/examples/fft.html)

- Beat Detection (also showing use of WebGL for 3D visualizations)

- Writing Audio from JavaScript, Digital Signal Processing
  - JavaScript Text to Speech engine
    - [http://scotland.proximity.on.ca/dxr/tmp/audio/tts/index2.html](http://scotland.proximity.on.ca/dxr/tmp/audio/tts/index2.html)
  - JavaScript Audio Sampler
    - [http://ware.buildingsky.net/processing/dsp.js/examples/sampler.html](http://ware.buildingsky.net/processing/dsp.js/examples/sampler.html)
  - Experimental JavaScript port Pure Data
  - Delay effect
  - Csound shaker instrument ported to JavaScript via Processing.js
    - [http://scotland.proximity.on.ca/dxr/tmp/audio/shaker/instruments/shaker.html](http://scotland.proximity.on.ca/dxr/tmp/audio/shaker/instruments/shaker.html)
The Web Audio API (2010)

Proposed by Chris Rogers (ex Apple Core Audio Engineer) and his team at Google

Audio Graph concept with nodes' DSP part coded natively in the browser...
<table>
<thead>
<tr>
<th><strong>Web Audio Demos</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Playground</strong> (a,i)</td>
</tr>
<tr>
<td>The Web Audio Playground helps developers visualize how the graph nodes in the Web Audio API work. Shown at I/O 2012.</td>
</tr>
<tr>
<td><strong>Vocoder</strong> (a,i,m)</td>
</tr>
<tr>
<td>This complex audio processing app (shown at I/O 2012) implements a 28-band vocoder - a &quot;robotic voice&quot; processor.</td>
</tr>
<tr>
<td><strong>&quot;Analog&quot; Synth</strong> (a,m)</td>
</tr>
<tr>
<td>This application implements a polyphonic &quot;analog&quot; synthesizer, with a classic voice architecture. Playable via Web MIDI or onscreen keyboard.</td>
</tr>
<tr>
<td><strong>Input Effects</strong> (a,i)</td>
</tr>
<tr>
<td>This demo lets you play with a few common effects on the audio inputs.</td>
</tr>
<tr>
<td><strong>AudioRecorder</strong> (a,i)</td>
</tr>
<tr>
<td>This is sample code for recording audio from live input and downloading them as WAV files, built on Matt Diamond's excellent RecorderJS.</td>
</tr>
<tr>
<td><strong>Oscilloscope</strong> (a)</td>
</tr>
<tr>
<td>Oscilloscope waveform display for Web Audio, and pulse-width duty cycle control on a square wave.</td>
</tr>
<tr>
<td><strong>Monosynth</strong> (a,m)</td>
</tr>
<tr>
<td>Example of a monophonic Web MIDI/Web Audio synth, with no UI.</td>
</tr>
<tr>
<td><strong>wubwubwub</strong> (a,m)</td>
</tr>
<tr>
<td>This application implements a dual DJ deck, specifically intended to be driven by a Numark MIDI controller.</td>
</tr>
<tr>
<td><strong>Pitch Detection</strong> (i)</td>
</tr>
<tr>
<td>This application performs monophonic auto-correlating pitch detection in realtime. This is suitable for a guitar tuner or other complex waveform source.</td>
</tr>
<tr>
<td><strong>The Game of Life</strong> (m)</td>
</tr>
<tr>
<td>Conway's Game of Life, on an 8x8 grid, interfacing with a Novation Launchpad controller via Web MIDI.</td>
</tr>
<tr>
<td><strong>Happy Drums</strong> (a,m)</td>
</tr>
<tr>
<td>Quick addition of MIDI controller support (using a Livid Instruments CNTRLR) to the Shiny Drum Machine.</td>
</tr>
<tr>
<td><strong>TemplateSynth</strong> (a,m)</td>
</tr>
<tr>
<td>Simple template for Web Audio synthes, with a polyphonic voice architecture, on-screen keyboard (including touch), and Web MIDI.</td>
</tr>
<tr>
<td><strong>TouchPad</strong> (a,i)</td>
</tr>
<tr>
<td>Pointer/multi-touch input to control Web Audio. Load it on a touch device, touch and slide around - to open/close the filter and modify Q.</td>
</tr>
<tr>
<td><strong>Volume Meter</strong> (i)</td>
</tr>
<tr>
<td>This is a simple code example of how to properly implement a clip-detecting volume meter.</td>
</tr>
<tr>
<td><strong>Sampl</strong> (a,m)</td>
</tr>
<tr>
<td>This is an in-progress example of how to implement a sample-playback synthesizer. Eventually should support looping and voice control, but simple code example of how to properly implement a clip-detecting volume meter.</td>
</tr>
<tr>
<td><strong>Slide Deck</strong></td>
</tr>
<tr>
<td>Slide deck on Web Audio and Web MIDI.</td>
</tr>
</tbody>
</table>
Creation of a W3C Web Audio Working Group

Meeting every 1-2 weeks

In person meeting 2-3 times a year

(right picture from 2021)
Some milestones

15/12/2011: first draft specification of the WebAudio API

- Lot of work to make sure it can be implemented and sound the same on all browsers (algorithms, etc.)

2012:

- Implementations in Firefox/Webkit browsers
- Support for real time input: Media Capture and Streams API
- First draft of the WebMidi API, implementation in Google Chrome
  - (2022 in Firefox too!)
But… porting existing DSP code was not easy…

The WebAudio API was incomplete!

Need for something better!
Some milestones

2018-2022: AudioWorklet: custom DSP processing in the Audio Thread

2012: WebSockets (synchronous collaboration)

2015: Web Components

2018: WebAssembly
Recent history…

2021: the WebAudio API V 1.0 became a W3C Recommendation in June 2021 (aka a “frozen standard”)

2021-today: ongoing development of Version 2.0

- The working group took care of github issues/proposals
- Conducted a developers’ survey

2022: W3C proposes help for higher level standards (i.e. for a plugin standard)
Some notable demos
<table>
<thead>
<tr>
<th><strong>Playground</strong> (a,i)</th>
<th><strong>Vocoder</strong> (a,i,m)</th>
<th><strong>&quot;Analog&quot; Synth</strong> (a,m)</th>
<th><strong>Input Effects</strong> (a,i)</th>
<th><strong>AudioRecorder</strong> (a,i)</th>
</tr>
</thead>
<tbody>
<tr>
<td>The Web Audio Playground helps developers visualize how the graph nodes in the Web Audio API work. Shown at I/O 2012.</td>
<td>This complex audio processing app (shown at I/O 2012) implements a 28-band vocoder - a &quot;robotic voice&quot; processor.</td>
<td>This application implements a polyphonic &quot;analog&quot; synthesizer, with a classic voice architecture. Playable via Web MIDI or onscreen keyboard.</td>
<td>This demo lets you play with a few common effects on the audio inputs.</td>
<td>This is sample code for recording audio from live input and downloading them as WAV files, built on Matt Diamond's excellent RecorderJS.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th><strong>Oscilloscope</strong> (a)</th>
<th><strong>Monosynth</strong> (a,m)</th>
<th><strong>wubwubwub</strong> (a,m)</th>
<th><strong>Pitch Detection</strong> (i)</th>
<th><strong>The Game of Life</strong> (m)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Oscilloscope waveform display for Web Audio, and pulse-width duty cycle control on a square wave.</td>
<td>Example of a monophonic Web MIDI/Web Audio synth, with no UI.</td>
<td>This application implements a dual DJ deck, specifically intended to be driven by a Numark MIDI controller.</td>
<td>This application performs monophonic autocorrelating pitch detection in realtime. This is suitable for a guitar tuner or other complex waveform source.</td>
<td>Conway's Game of Life, on an 8x8 grid, interfacing with a Novation Launchpad controller via Web MIDI.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th><strong>Happy Drums</strong> (a,m)</th>
<th><strong>TemplateSynth</strong> (a,m)</th>
<th><strong>TouchPad</strong> (a,t)</th>
<th><strong>Volume Meter</strong> (i)</th>
<th><strong>Samplr</strong> (a,m)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Quick addition of MIDI controller support (using a Livid Instruments CNTRLR) to the Shiny Drum Machine.</td>
<td>Simple template for Web Audio synths, with a polyphonic voice architecture, on-screen keyboard (including touch), and Web MIDI.</td>
<td>Pointer/multi-touch input to control Web Audio. Load it on a touch device, touch and slide around - to open/close the filter and modify Q.</td>
<td>This is a simple code example of how to properly implement a clip-detecting volume meter.</td>
<td>This is an in-progress example of how to implement a sample-playback synthesizer. Eventually should support looping and voice control, but simple code example of how to properly implement a clip-detecting volume meter.</td>
</tr>
</tbody>
</table>

**2011-2015: Chris Wilson’s demos!**
2014: first implementation of ToneJS library

Used in TONS of projects!
2015: Web Audio debugger in Firefox
WAM SYNTHS

WebAssembly / AudioWorklet

.asm.js / ScriptProcessor

2015: Web Audio Modules (Kleimola & Larkin)
2015: first commercial DAWS

- Soundtrap.com (spotify)
- Bandlab.com
- Ampedstudio.com
- Soundation.com
2016: why not some guitar?
2014/2015: DSLs compile to ASM.js and later to WebAssembly

FAUST, PureData, Csound, others can be run on the Web

2018: FAUST online IDE
2018: Ableton comes in the game!
2018: the live coding community shines

Strudel (Roos, Mclean), Gibber, (Roberts and Joann), Estuary (Ogborn and Beverley), Hydra, Wags, Feedforward, etc.

Strudel in 2022

Some projects started in 2012...
2022: Web Audio plugins become mature (WAM)
2022: Roland comes in the game!
2022: Microtonal synthesizer (Khyam Allami / counterpoint)

**LEIMMA**: explore microtonial tuning systems

**APOTOME**: generative music composition tool, which lets you use all the tunings from LEIMMA

We’ve just launched a family of projects that explore the world of microtonal tuning systems, developed in collaboration with @KhyamAllami over the past two years.

Traduire la Tweet
2022: Audacity (the real one) on the Web thanks to WASM

Port of **audacity on the Web!**

- WASM compiled

Also: WASM port of **transitionDJ app**
CONCLUSION

The Web platform is Magic :-)  

Web Audio is a technology but it also became a new, multidisciplinary, research field

Web Audio lives along with all other APIs, and goes beyond the browser now…

WebML, WebCodecs, WebAudio API standalone in embedded systems, etc.