Interactive Web Audio Networking System and Method with DrSax.js

Euyshick Hong
MARTE Lab.
Dongguk University
Jung-gu Pil-dong, Seoul, Korea
antaresax@dongguk.edu

Jun Kim
MARTE Lab.
Dongguk University
Jung-gu Pil-dong, Seoul, Korea
music@dongguk.edu

ABSTRACT

Web technologies have recently been developed by software engineer and Information Technology companies. Especially, web audio applications with web audio libraries and frame-works have studied for synthesizing and audio processing using Web Audio API on JavaScript. JavaScript is a flexible programming language with a front side and server side capabilities, providing dynamic interactions and networking system with the web. This paper aims an Interactive web audio networking system using a node.js and web audio library DrSax.js through a Web Audio API for artistic expression and various musical applications.

CCS Concepts

•Applied computing → Arts and humanities → Sound and music

Keywords

Web audio API; network; node.js; socket.io; JavaScript.

1. INTRODUCTION

In last few years, Web technologies have significantly developed with the advent of HTML5 and JavaScript [1], [2], including a number of libraries, frameworks and novel web audio applications and networking system that take advantage of web audio API [3], [4] and the server side language that node.js [5].

Novel web audio applications have built with web audio API for audio processing, sound effects, synthesizing, visualization, and media control for web based applications. Also, several of interactive networking systems using web audio API has been developed with node.js of JavaScript. For example, Play the light of Monet [6] presents interactive networking works, such as web audio ensembles where users can connect each other's playing real-time at the same location, or spatially distant, using web applications instrument to tablets, smart phones, laptops, or desktop.

Users access the web audio application through the WI-FI or 3G, 4G data. Such methods for musical networking system can be built using various web audio API and libraries. Many web audio frameworks and libraries have been released to extend web audio applications, and new various features have been provided.

ICCIP'17, November 24-26, 2017, Tokyo, Japan.
© 2017 Association for Computing Machinery.
ACM ISBN 978-1-4503-5365-6/17/11…$15.00
DOI:

In this paper aims to find interactive web audio networking system through a node.js and DrSax.js that is a web audio library optimized for versatile web audio works. Using DrSax.js have the advantage of helping to simply and easily to develop new web audio applications.

Commonly, web audio applications require data controls and visualization to enable multiple interactions, such as on/off buttons, slides, dials and pitch/ volume values. DrSax.js supports these extra functionalities. Thus, it is necessary to use an optimized web audio library and framework that includes visualization and provides comprehensive features. DrSax.js is provided as a web audio library and framework, making it possible for users to easily and quickly build web audio applications. Consequently, this paper has proposed following systems.

- To build an interactive networking system with node.js in JavaScript for real-time communication.
- To use a DrSax.js for sound processing and effects.
- To connect at web audio application with a mobile or computer through WI-FI or 3G/4G data easily in each different place

This study has presented an Interactive web audio networking system using JavaScript technologies for musical expression and various multimedia works.

2. RELATED WORKS

There are various web libraries and frameworks in JavaScript for web audio works and multimedia system. Interface.js

[7] enables control of web interactive system through a server interface to mouse, touch screen, and portable de-vices with Gibberish.js [7] that is a sound processing library for audio synthesizers and sound effects. WAAX [8], [9], Audiolib.js [10] and Tuna.js [11] also are a audio processing library for web audio works. Some other visual libraries and JavaScript technologies have taken different approaches. Extree.js [12] is a JavaScript 3D visual library that can facilitate simple interaction for web visual application and media art installation. WebGL [13] was developed for 2D/3D visual control, providing a cross-platform frame-work for 3D graphics based on OpenGL ES 2.0, exposed through the HTML5 Canvas elements using JavaScript.

3. DESIGN CONCEPT OF SYSTEM

The design concept of this system was to provide a useful interactive web audio networking system through Dr-SAX.js for sound processing and node.js for asynchronous non-blocking networking system.

3.1 Utilization of DrSAX.js

3.1.1 Sound Units

DrSax.js audio units can be separated into four parts that synthesizer, audio file control, sound input and sound effects. Synthesizers consist of Amplitude modulation, Frequency modulation and subtractive modulation and basic oscillators.

These synthesizers provide not only basic functions, but also sound processing to connect input sources, such as audio files, a microphone. Sound input units include basic microphone input and tuning frameworks. Sound effect units provide sound processing elements, such as equalizers, compressor and delay, reverb, and stereo pan. These are mainly effects elements in sound units frameworks.

Figure 1 shows a simple subtractive modulation synthesizer application with DrSax.js https://drsax.github.io/DrSAX/examples/demo_fm.html

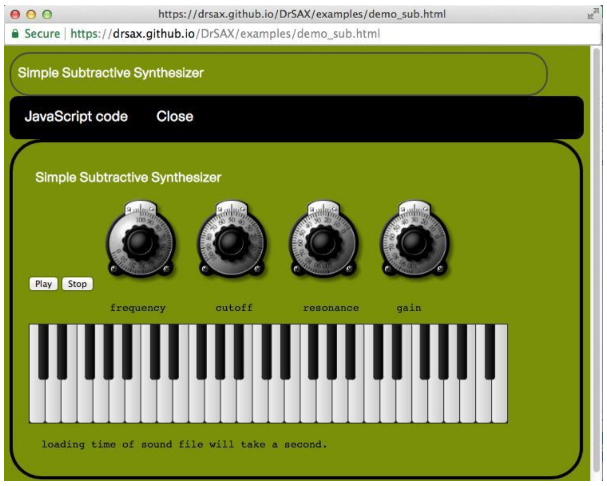
https://drsax.github.io/DrSAX/examples/demo_sub.html
Secure | https://drsax.github.io/DrSAX/examples/demo_sub.html
Simple Subtractive Synthesizer
JavaScript code
Close
Simple Subtractive Synthesizer
Play
Stop
frequency
cutoff
resonance
gain
loading time of sound file will take a second.

Figure 1. Simple subtractive synthesizer

3.1.2 Sound Visualization and Data Control

Sound visualization is a useful function to monitor in pitch, volume and sound timbre without sound outputs. DrSax.js provides frequency and amplitude domain sound visualization. The HTML5 has Canvas API that is web based visualization canvas control for visuals, as well as sound input and visualization.

Figure 2 shows a sound visualization through sound input. https://drsax.github.io/DrSAX/examples/ demo_mic.html

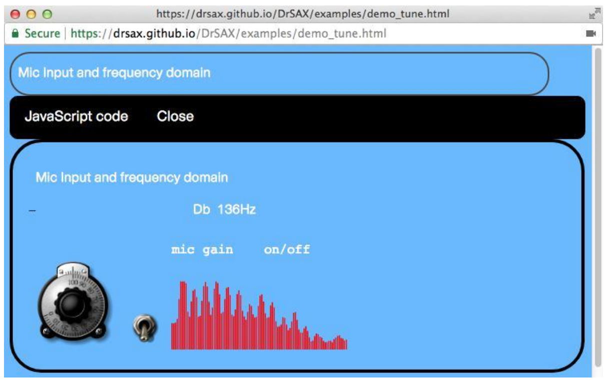
https://drsax.github.io/DrSAX/examples/demo_tune.html
Secure | https://drsax.github.io/DrSAX/examples/demo_tune.html
Mic Input and frequency domain
JavaScript code
Close
Mic Input and frequency domain
Db 136Hz
mic gain
on/off

Figure 2. Sound input and visualization

Two types of data control units have designed for real-time control of different styles. First, Data control API can control volume, frequency, modulation parameters and volume/pitch. Second, toggle API enables control of on/off frameworks in the web audio application.

3.1.3 DrSax.js tutorial site

The Dr.Sax.js tutorial provides a simple web audio applications, simple code and detail information. http://drsaxjs.cf

3.2 Networking Structure

Most of a web client requests some data to server side and server side responds requested data to the web client. But almost server side system is not non-blocking I/O system. So, we have used socket.io[3] module in node.js server frame-work in JavaScript.

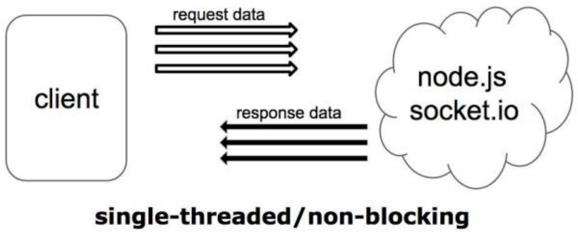

Figure 3. node.js networking system flow

3.2.1 Node.js and socket.io Module

With the development of web server side, non-blocking I/O system has developed between the web client and server. The system can be used interactive web networking systems. So, this project has developed with Node.js server frameworks and Socket.io [14] server module. Node.js is the asynchronous non-blocking networking system in real-time by JavaScript. Socket.io is a module that can communicate two-way in real-time with node.js, such as message networking system on mobile or PC. Figure 3 shows asynchronous non-blocking I/O system of node.js.

4. PROGRAMMIG

4.1 Sound Processing

4.1.1 Implementation with DrSAX.js

```
1  <script src="DrSax.js"></script>
2  <script>
3
4    var DSX = new DSX;
5    var osc = new DSX.Osc({type:"sine",
         freq:700});
6    var am = new DSX.AM({ mod_type :"
         sine",
7        modfreq:200, depth:0.5, gain:0.5
             });
8
9    am.get(osc);
10   am.connect(gain);
11   gain.connect(DAC);
12   osc.start();
13
14 </script>
```

Figure 4. Using DrSax.js

To use DrSax.js in web audio applications, download Dr-Sax.js file from the tutorial site and declare in a script tag. Line 4 in Listing 1, then declare "var DSX = new DSX" on the top. Listing 1 shows an example of simple Amplitude modulation (AM) synthesizer. Line 5 shows a sound oscillator frequency and type.

[3] Socket.IO: real-time bidirectional event-based communication module.

The AM synthesizer has 4 data values and declares from 6 to 7 that frequency, modulator, depth, and gain parameters. Line 9 shows to connect an oscillator and AM to gain. Lines 12 shows to play.

4.1.2 *Sound Input and Visualization*

Sound Visualization depends on pitch and sound volume that to get an input source. Listing 3, the mic input source constructs in line 3 have initial values set by object literals. From 2 to 4 show to set mic input and sound visualization, with line 4 setting a color based on the received mic input data, enabling interactive sound visualization using the HTML5 canvas API. The sound balance can be mixed during playing or recording. Lines 7 and 8 show to connect mic input source to gain and sound out.

```
1 var DSX = new DSX;
2 var amp = new DSX.Amp({gain: 0.8});
3 var micInput = new DSX.Mic();
4 var frequrncy_canvas = new DSX.freqDomain
                            ('canvas', 'black');
5 frequrncy_canvas.getAnalyser(amp);

7 micInput.connect(amp);
8 amp.connect(DAC);
```

Figure 5. Mic input and visualization

4.1.3 *Effects Data Control*

Listing 3, lines 1 to 4 shows to set a 700Hz sine oscillator with amplitude gain value 0.7. Line 5 shows to set delay effecter and data values, such as delay time and feed-back. Lines 8–10 show to connect osc to gain to delay to DAC (sound output). The DrSax.js includes various sound effects: delay, reverb, panning, compress and equalizer that can mix each sound effects to expand sound processing.

```
1 var gain = new DSX.Amp({gain: 0.7});
2 var osc = new DSX.Osc({
3         type:"sine",freq: 700
4 });
5 var Delay = new DSX.Delay({
6         delayTime :400, feedback: 0.7,
7 });
8  osc.connect(gain);
9  gain.connect(Delay);
10 Delay.connect(DAC);
```

Figure 6. Effects values

4.2 Networking with Node.js and Socket.io

4.2.1 *Client Side System*

```
1 var socket = io.connect();
2 socket.on('line', function(data) {
3     context.lineWidth = data.width;
4     context.strokeStyle = data.color;
5     context.shadowBlur = data.blurWidth;
6     context.shadowColor = data.blurColor;
7     context.lineCap = "round";
8     context.beginPath();
9     context.moveTo(data.x1, data.y1);
10    context.lineTo(data.x2, data.y2);
11    context.stroke();
12 });
```

Figure 7. Networking with socket.io

Web client receives a data from server side. Listing 3, socket declares at line 1. From line 2 to 12, socket received data from a socket.io of server side.

4.2.2 *Server side system*

```
1 var express = require('express');
2 var app     = express();
3 var http    = require('http').Server(app);
4 var io      = require('socket.io')(http);
5 var path    = require('path');
6 app.use(express.static(
          path.join(__dirname,"public")));
7 var port = process.env.PORT || 8358;
8 http.listen(port, function(){
9 console.log("Dr.Hong server on!"); });
10 io.on('connection', function(socket){
11 socket.on('draw', function (data) {
12          io.sockets.emit('line', data);
13 });
```

Figure 8. Using Socket.io and Node.js

Listing 5 shows a node.js server and socket.io networking that declares line 4. From line 11 to 13, it presents to communicate with the socket.io module and web client side. At line 11, socket.io received data from a web client and return to web client by line 12. it enables interactive networking system in real-time between server side and client side.

5. PERFORMANCE WORKS

5.1 Interactive Networking Performance System

Interactive web audio network system has developed web audio application (Dr.Saxoman) with DrSax.js and a hybrid saxophone interface (Dr.Saxophone II) in Figure 4 and 5, as shown in the Dr.Saxophone II can control the sound processing data to Dr.Saxoman in real-time that interactive networking system. This Interactive web audio network system was performed for an interactive performance " A White Night with Dr.Saxoman and Dr. Saxophone II" on stage by Euy Shick Hong in Lee Hae Rang theater [4]

5.1.1 *Dr.Saxoman*

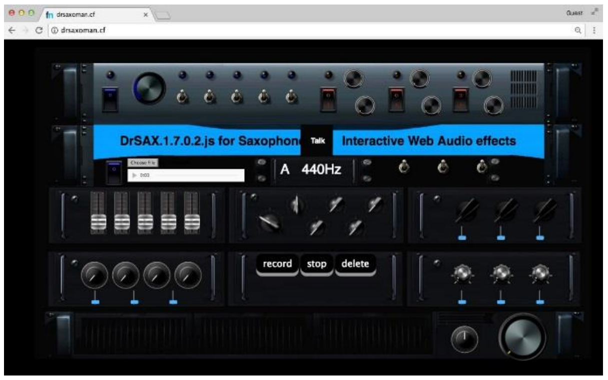

Figure 9. DrSaxoman: web audio application

Dr.Saxoman is a web audio application for hybrid saxophone interface or wind acoustic instrument and sound input that supports sound tuner, visualization, audio file, effects, recording, equalizer, compressor, AM, pitch shift, stereo reverb, delay, and stereo pan. All data values can be controlled a mouse or Dr.

[4] https://youtu.be/THJLy0GvAso

Saxophone II in real time. Also, Dr.Saxoman can be controlled to other wind instruments or voice for various interactive works.

- Synthesizer: FM, AM, subtractive.
- Sound input: sound input, tuner.
- Effect: eq, delay, compressor, pitch shift, reverb, pan.
- Interactive system: netwoking with DrSaxophone II.

5.1.2 *Dr.Saxophone II*

Dr.Saxophone II is a hybrid saxophone interface to control

web audio application in a interactive performance that expanded Dr.Saxophone. Dr.Saxophone(https://youtu.be/ do_yuLdVTCo)is controlled by integrated interface and lift up/down with saxophone.

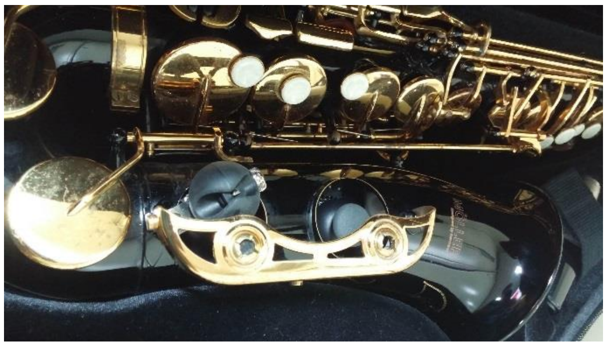

Figure 10. drsaxophone II

5.2 Media Art Installation

5.2.1 *Composition 2017*

Composition 2017 facilitates interactive web audio/drawing networking system using DrSAX.js and node.js with mobile and desktop PC for media art installation. composition 2017(https://vimeo.com/223936815) shows drawing visuals and sounds y mobile web drawing application. Each user can communicate through other user's drawing and sound processing in real-time at same space or different place. Also, users can easily access the web drawing application through WI-FI or 3G, 4G data. Figure 6 shows the composition 2017: interactive web audio/drawing net-working system.

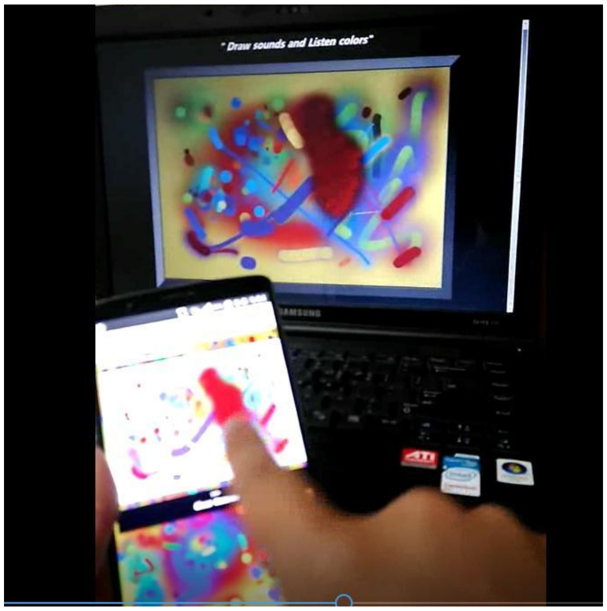

Figure 11. Web audio/drawing media installation

6. CONCLUSIONS

The Interactive web audio networking system has developed with web technologies that audio libraries and server side frameworks using JavaScript. Especially, web audio libraries have released for web audio works using Web Audio API. Also, JavaScript technologies are expanding server side field that node.js. This paper described a web audio library and interactive networking system to control real-time audio processing, visualization, sound effects, and media installation. This paper aims an Interactive web audio net-working system using web audio library DrSax.js through a Web Audio API and server framework node.js for artistic expression and interactive web musical applications.

7. TUTORIAL AND WEB APPLICATION SITE

DrSAX.js tutorial and Github site.

- DrSax.js tutorial site:
  http://drsaxjs.cf
- DrSax.js link in Github:
  https://drsax.github.io/DrSAX/DrSax.1.7.0.1.2. js
- DrSaxoman application site :
  http://drsaxoman.cf
- Composition2017 : http://ismarte.cf
  (installation app) http://marte.cf (mobile app)

8. REFERENCES

[1] Web audio tutorials -middle ear. Middle ear: http://middleearmedia.com/category/web-audio-tutorials/.

[2] Tutorial of javascript,html. css. Development. Technology Training Center: https://www.developphp.com/.

[3] C. Rogers, web audio api-w3c working draft. Web audio API:https://www.w3.org/TR/webaudio.

[4] Web audio api - mozillawiki. Mozilla: http://wiki.mozilla.org.

[5] Node.js. Node.js= https://nodejs.org.

[6] Play the light of monet: interactive web audio networking system. youtube link= https://youtu.be/uZLOY8onwz4.

[7] C. Roberts, G. Wakefield, and M. Wright. The web browser as synthesizer and interface. in Proceedings of the 13th Conference on New Interfaces for Musical Expression(NIME-13), Daejeon, Korea,, 2013.

[8] H. Choi and J. Berger. Waax: Web audio api extension. in Proceedings of the 13th Conference on New Interfaces for Musical Expression (NIME-13), Daejeon, Korea,, 2013.Web audio api extension.

[9] Web audio api extension. Github code: https://github.com/hoch/waax.

[10] A powerful audio tools library for javascript. Audio.js: https://github.com/jussikalliokoski/audiolib.js.

[11] An audio effects library for web audio. Tuna.js: https://github.com/Theodeus/tuna.

[12] Javascript 3d library. Tree.js: https://github.com/mrdoob/three.js/.

[13] Webgl - opengl es 2.0 for the web. Webgl: https://www.khronos.org/webgl/.

[14] Socket.io. socket.io site: http://socket.io.