

# Creative Coding School in Sapporo

Advanced

2015.8.9

JavaScript Sound Programming

# Web Audio API

15:00 説明

15:30 ハンズオン

16:30 休憩

17:00 作品制作

18:00 作品発表

18:30 終了

# Web Audio API


# Sound Programming

今日の資料とプログラムはここからダウンロードできます

<https://github.com/aike/ccss2015/>

# Web Audio API

W3C Working Draft



Web Audio API

W3C Working Draft 10 October 2013

**This Version:**  
<http://www.w3.org/TR/2013/WD-webaudio-20131010/>

**Latest Editor's Draft:**  
<http://webaudio.github.io/web-audio-api/>

**Latest published version:**  
<http://www.w3.org/TR/webaudio/>

**Previous version:**  
<http://www.w3.org/TR/2012/WD-webaudio-20121213/>

**Editors:**  
Paul Adenot, Mozilla Foundation  
Chris Wilson, Google  
Chris Rogers, Google (until August 2013)

**Authors:**  
See [Acknowledgements](#)

Copyright © 2013 W3C® (MIT, ERCIM, Keio, Beihang). All Rights Reserved. W3C [liability](#), [trademark](#) and [document use](#) rules apply.

Abstract

This specification describes a high-level JavaScript API for processing and synthesizing audio in web applications. The primary paradigm is of an audio routing graph, where a number of [AudioNode](#) objects are connected together to define the overall audio rendering. The actual processing will primarily take place in the underlying implementation (typically optimized Assembly / C / C++ code), but [direct JavaScript processing and synthesis](#) is also supported.

(typically optimized Assembly / C / C++ code), but [direct JavaScript processing and synthesis](#) is also supported. This specification describes a high-level JavaScript API for processing and synthesizing audio in web applications.

<http://www.w3.org/TR/webaudio/>

# Web Audio API

- ・ 本格的なサウンド表現が手軽にできる
- ・ 作ったものをすぐにシェアできる
- ・ ウェブ技術資産とシームレスにつながる  
Web API、Canvas、WebGL、加速度センサー、etc..
- ・ スマホでも動く  
一部制限あり

# Web Audio API

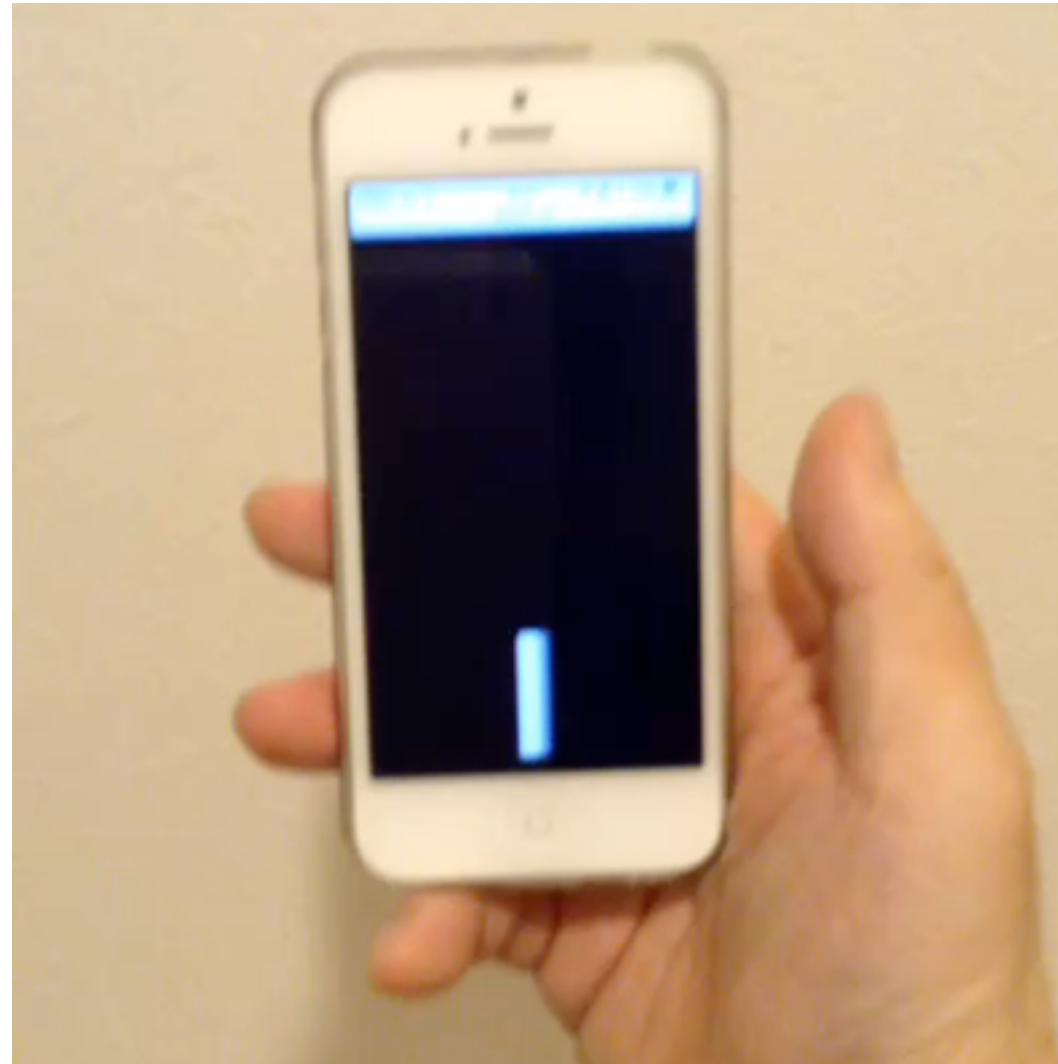
## Twitter APIとの連携



<http://beatonica.com/>

# Web Audio API

## 加速度センサーとの連携

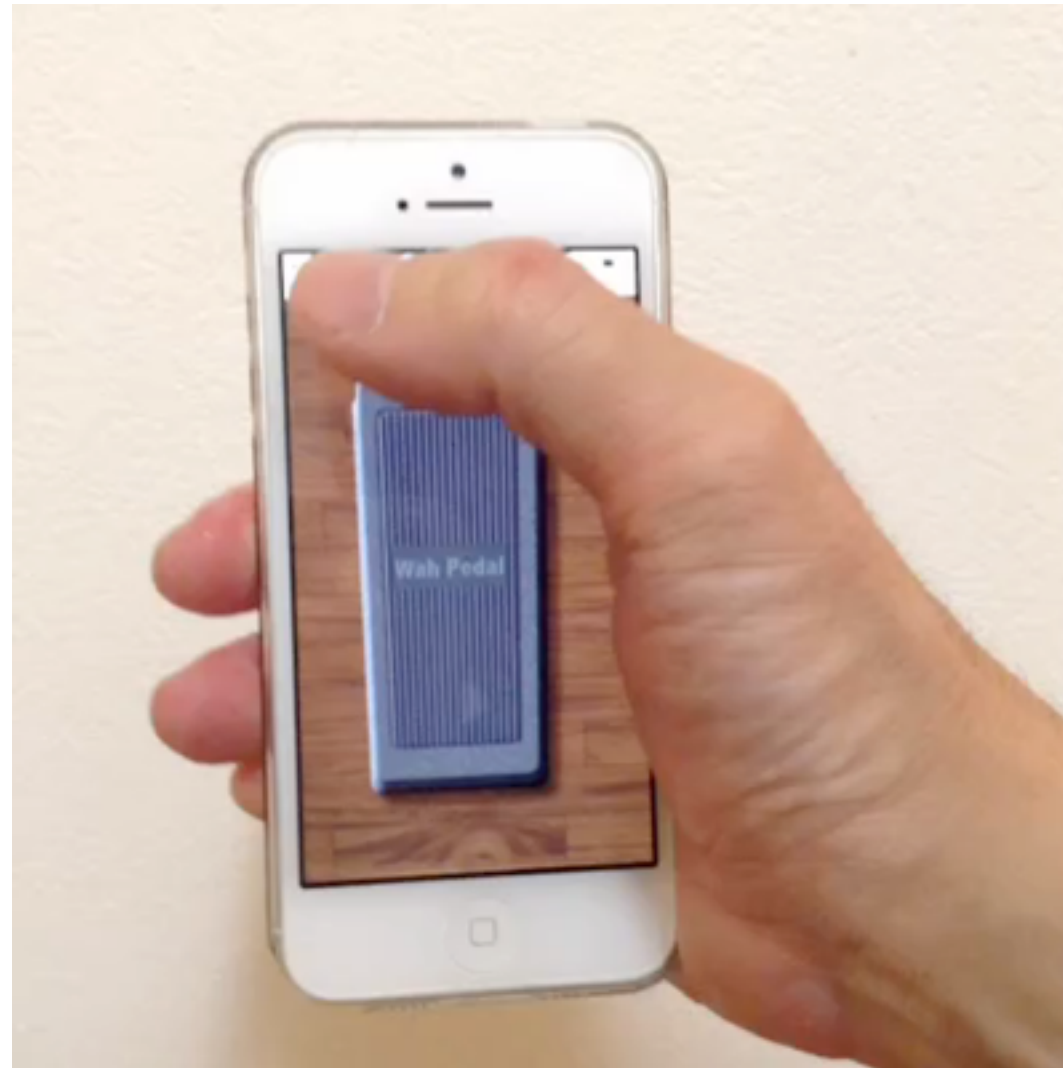


<http://aikelab.net/sw/>



# Web Audio API

加速度センサーとの連携



<http://aikelab.net/iphonewah/>

# Web Audio API

## WebGLとの連携



<http://aikelab.net/websynthv2/>

# Sound Programming

# Sound Programming

できるだけコピー＆ペーストを活用して入力してください

<https://github.com/aike/ccss2015>  
/advanced

# 1\_keyboard キーボード入力で演奏

## onkeydown / onkeyup

```
document.onkeydown = function(e) {  
    // キーコード49「1」が押されたらノートナンバー60(ド)  
    var note_no = e.keyCode - 49 + 60;  
    // ノートナンバーから周波数を計算  
    freq = 440.0 * Math.pow(2.0, (note_no - 69.0) / 12.0);  
    noteon();  
}  
// キーボードを離すと音を止める  
document.onkeyup = function(e) {  
    noteoff();  
}
```

## 2\_delay ディレイエフェクト

### Delay

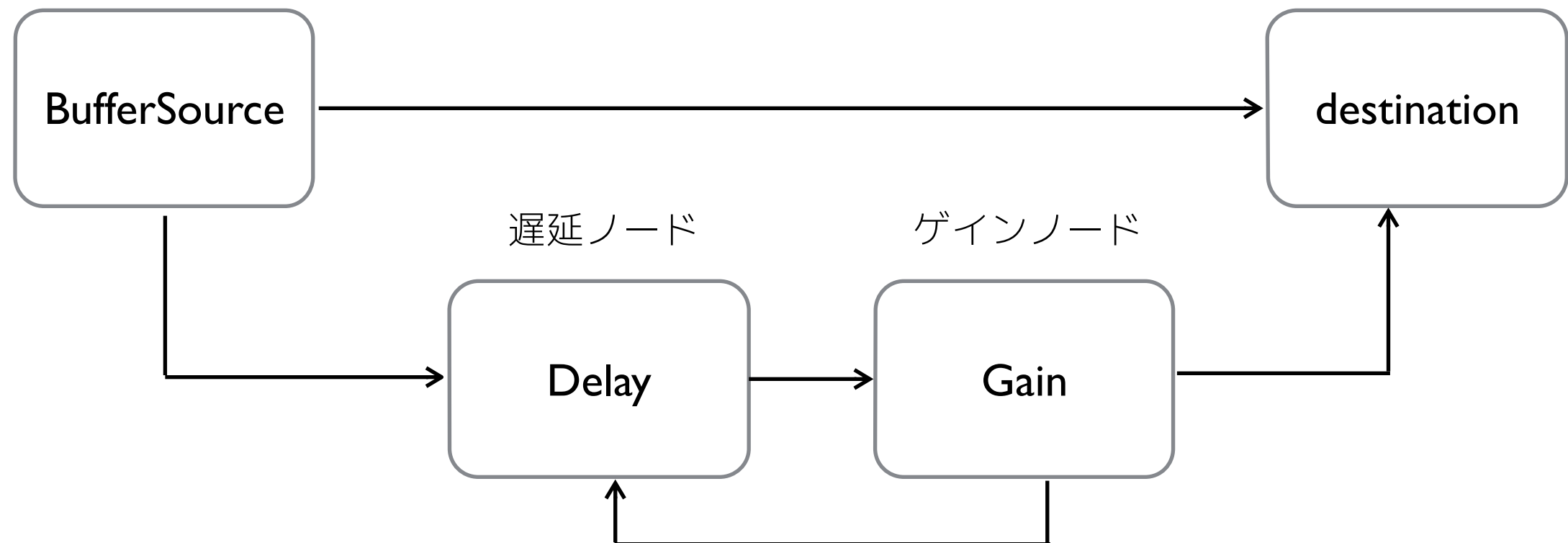
```
// フィードバック付ディレイエフェクト  
var delay = ctx.createDelay();  
delay.delayTime.value = 0.4; // 400ミリ秒  
var feedback = ctx.createGain();  
feedback.gain.value = 0.35; // 35%フィードバック  
delay.connect(feedback);  
feedback.connect(delay);  
feedback.connect(ctx.destination);
```

# 2\_delay ディレイエフェクト

## Delay

サウンド再生ノード

出力先ノード



# 3\_rhythmmachine リズムマシンの作成

## 複数サウンドファイルの取得

```
// サウンドファイルの取得
function getwav(name, callback) {
  fetch(name).then(function(res) {
    return res.arrayBuffer(); }).then(function(arr) {
    ctx.decodeAudioData(arr, function(buf) {
      callback(buf);
    });
  });
}

var kick, hat, snare, clap;
getwav("kick.wav", function(buf){kick = buf;});
getwav("hat.wav", function(buf){hat = buf;});
getwav("snare.wav", function(buf){snare = buf;});
getwav("clap.wav", function(buf){clap = buf;});
```



## 3\_rhythmmachine リズムマシンの作成

### setInterval

```
// 200ミリ秒ごとにplaysound関数を実行  
timer = setInterval(playsound, 200);
```

```
function playsound() {  
  if (beat % 2 == 0) { // 2拍ごとに実行  
    var wav = ctx.createBufferSource();  
    wav.buffer = kick;  
    wav.connect(ctx.destination);  
    wav.start();  
  }  
  beat = (beat + 1) % 8; // 0~7を繰り返す(8分音符)  
}
```

# 4\_response 音に反応してビジュアライズ

## CSS

```
#draw {  
  position: absolute;  
  font-weight: 700;  
  font-family: serif;  
  /* colorとtext-shadowで光のにじみの表現 */  
  color: #bbc;  
  text-shadow: 0 0 45px #44a, 0 0 90px #77e;  
}
```

## 4\_response 音に反応してビジュアライズ ScriptProcessor

```
var sensor = ctx.createScriptProcessor(1024, 1, 1);  
// 1024サンプルごとに呼ばれる処理 (1024/44.1kHz=23.2msec)  
sensor.onaudioprocess = function(event) {  
    var sin = event.inputBuffer.getChannelData(0);  
    var level = 0;  
    for (var i = 0; i < sin.length; i++) {  
        // 1024サンプルごとに波形の絶対値(音量)を合計  
        level += Math.abs(sin[i]);  
    }  
    showChar(level); // 23.2ミリ秒毎に音量を引数として関数を呼ぶ  
};
```

# 5\_jquery jQueryプラグイン

## jQuery

- ・ JavaScriptの汎用ライブラリ
- ・ プラグインフレームワークとして利用が多い
- ・ 最近のウェブフレームワークでは使わないこともあるが、デザイン系プラグインの数が圧倒的でまだまだ便利

# 5\_jquery jQueryプラグイン

## HTMLエレメントを震わせるプラグイン

### jRumble

The screenshot shows the official website for the jRumble jQuery plugin. The page has a clean, modern design with a blue header and a red 'Download' button. The left sidebar contains navigation links for 'About', 'Usage', 'Demos', 'Documentation', 'Contact', and 'Discussion'. The main content area is divided into sections: 'About' (describing the plugin's functionality), 'The Plugin' (a brief description), 'The Author' (introducing Jack Rugile), and 'Like the Plugin?' (encouraging support). A pink warning box is also present, advising users to read the documentation before using the plugin. At the bottom, there are social media sharing buttons for Twitter and Facebook, along with a 'Donate' button.

**jRumble<sup>v1.3</sup>**

- About
- Usage
- Demos
- Documentation
- Contact
- Discussion
- Download**

• Latest Version: 1.3  
• Latest Release Date: December 3, 2011  
• Original Release Date: March 30, 2011  
• Compressed: 1.47kb  
• Uncompressed: 4.84kb  
• [View GitHub Repository](#)

**About**

**The Plugin**

jRumble is a [jQuery](#) plugin that rumbles, vibrates, shakes, and rotates any element you choose. It's great to use as a hover effect or a way to direct attention to an element.

Please [read this](#) before using jRumble. Flashing and flickering objects on the web can be dangerous. Please use this plugin responsibly.

**The Author**

My name is [Jack Rugile](#). I am a web designer/developer living in Denver, Colorado. Feel free to follow me on Twitter: [@jackrugile](#).

**Like the Plugin?**

This plugin is free to use, however, if you enjoy jRumble and want to show some support, feel free share it or make a donation. It would be greatly appreciated!

[Donate](#) [Tweet](#) 1,546 [いいね!](#) 537

<http://jackrugile.com/jrumble/>

# 6\_canvas マウスとCanvasの使用

## Canvas

```
<canvas id="drawarea" width="1000" height="700"></canvas>
```

```
gtx.fillStyle = this.col;  
// shadowを設定してぼかしをかける  
gtx.shadowColor = this.col;  
gtx.shadowBlur = 50;  
// 描画  
gtx.beginPath();  
gtx.arc(this.x, this.y, this.r, 0, Math.PI * 2, true);  
gtx.fill();
```

# 6\_canvas マウスとCanvasの使用

## onclick

```
var circles = [];  
canvas.onclick = function(e) {  
    // 画面クリックすると円を追加  
    circles.push(new Circle(e.clientX, e.clientY, Math.random()));  
};
```

# 6\_canvas マウスとCanvasの使用

## requestAnimationFrame

```
// 描画ループ
function loop() {
  // 60FPSのタイマー
  requestAnimationFrame(loop);
  // 画面消去
  gtx.clearRect(0,0, gtx.canvas.width, gtx.canvas.height);
  // すべての円を再描画
  for (var i=0; i < circles.length; i++) {
    circles[i].draw();
  }
}
loop();
```



# 作品制作

音の出る作品であること

18:00 終了

## 【参考】

- ・ 今日のプログラムのいずれかを元に改変
- ・ 自分の得意なWeb技術と組み合わせてみる