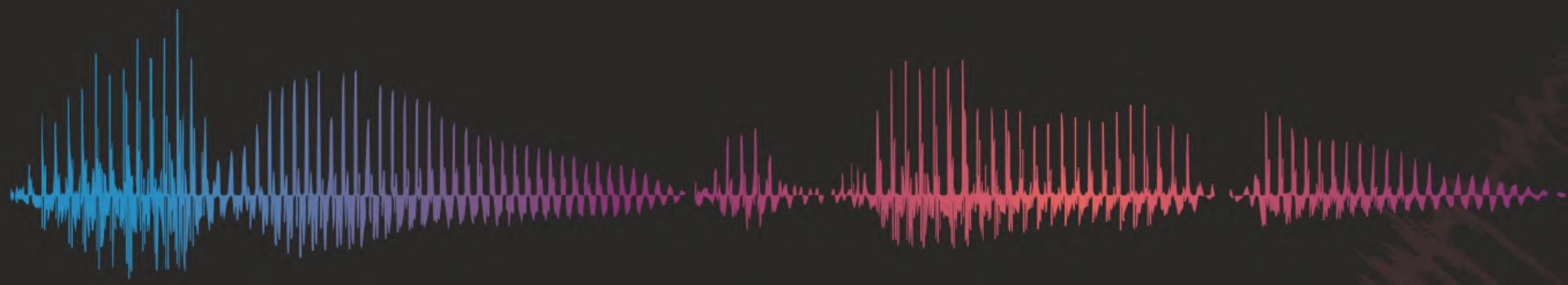


ADC<sup>21</sup>



# REAL-TIME REMOTE JAMS WITH WEBRTC AND WEB MIDI

PHILIP MILLER



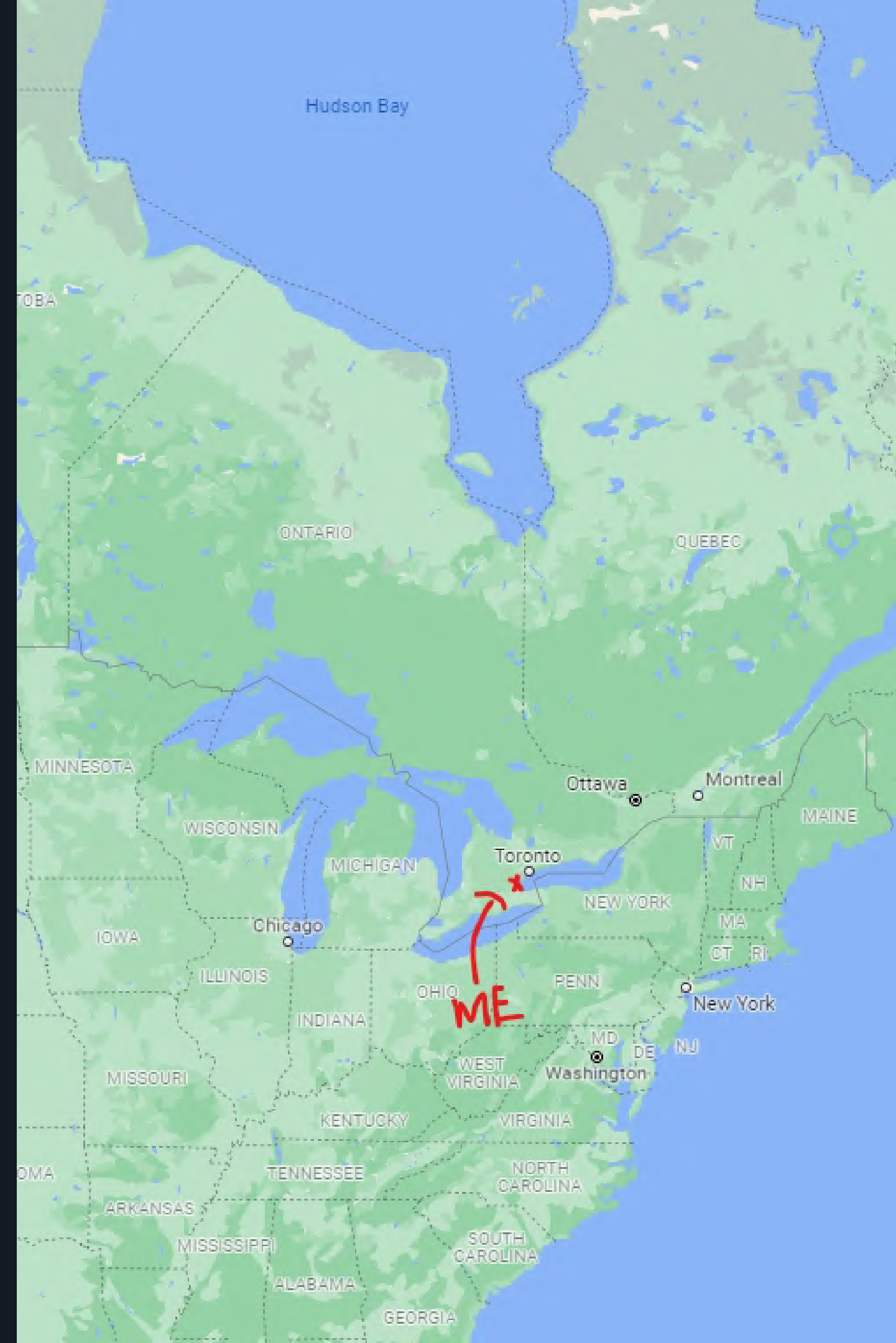
# Real-time remote jams with WebRTC and Web MIDI

*aka WebMIDI RTC*



# About me

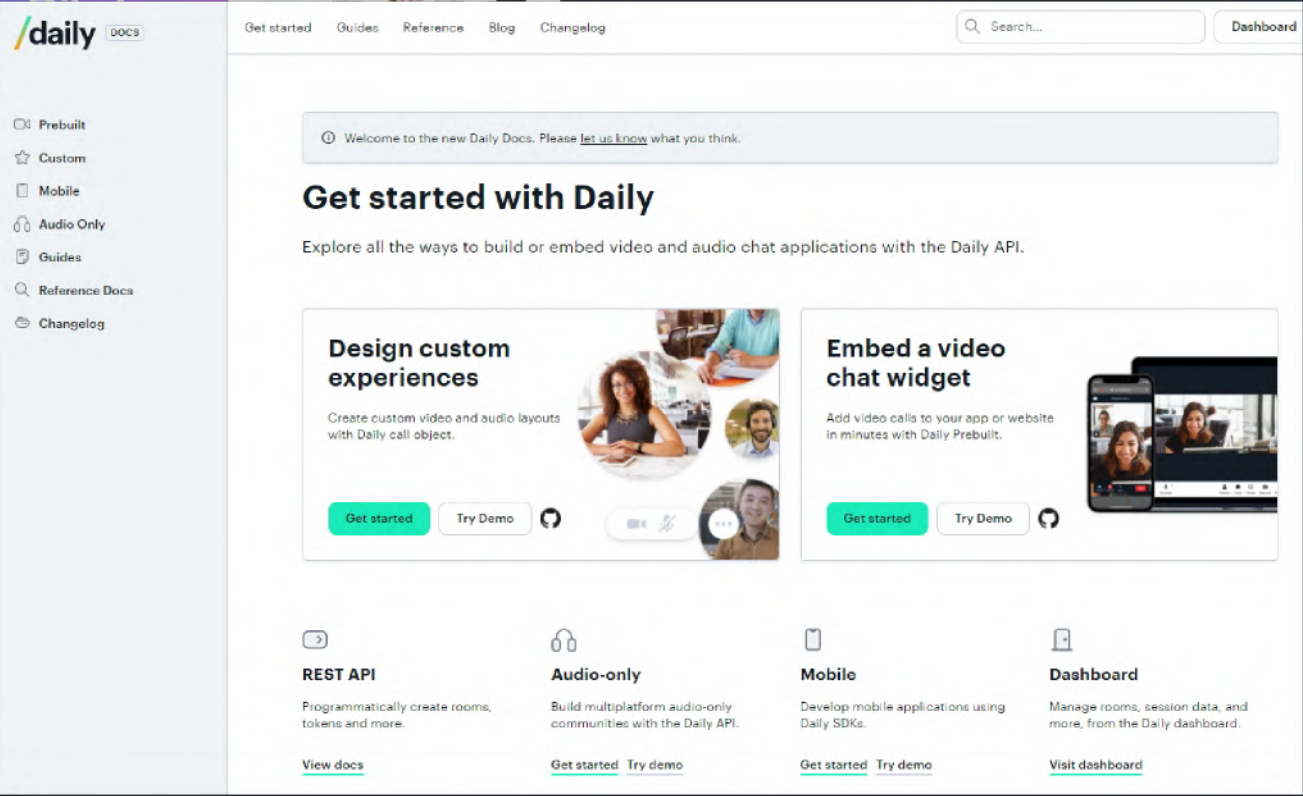
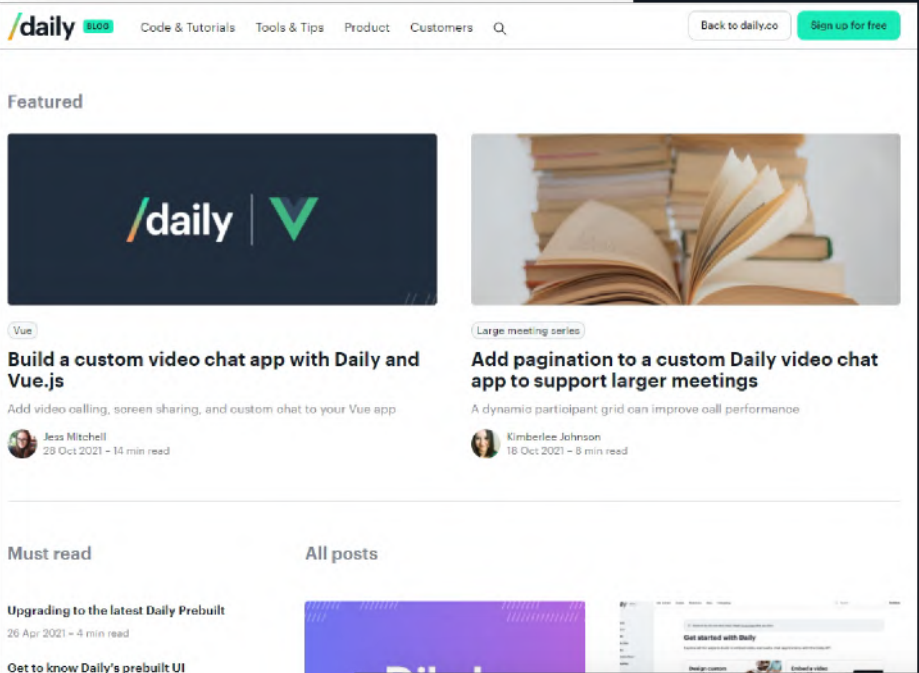
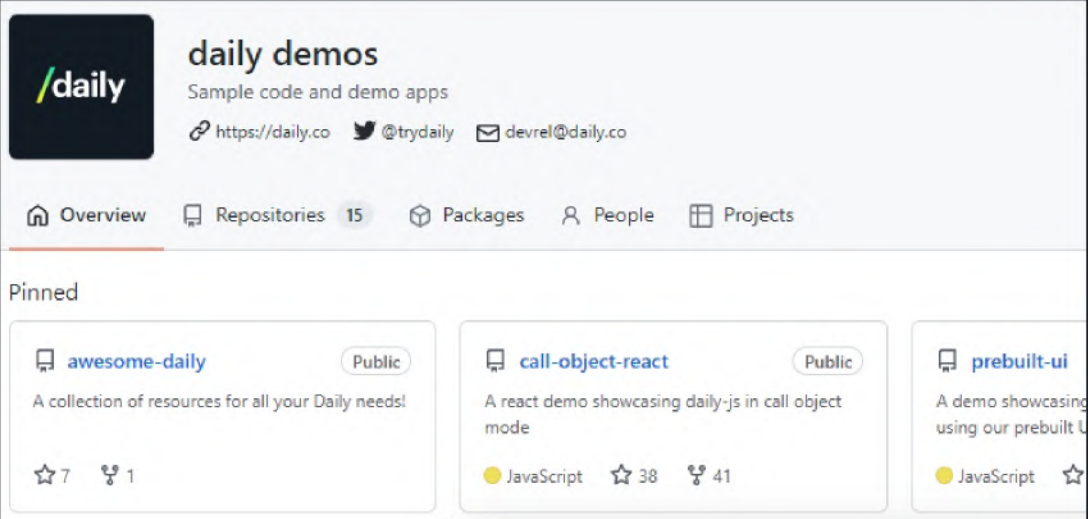
- Located in Hamilton, ON, Canada





# About me

- Sr. DevRel Engineer @ Daily





# About me

- Drummer turned synth nerd



**The web is coming of age**  
and we're finally starting to reap the benefits

**Video calls**



# 1964



So elegant

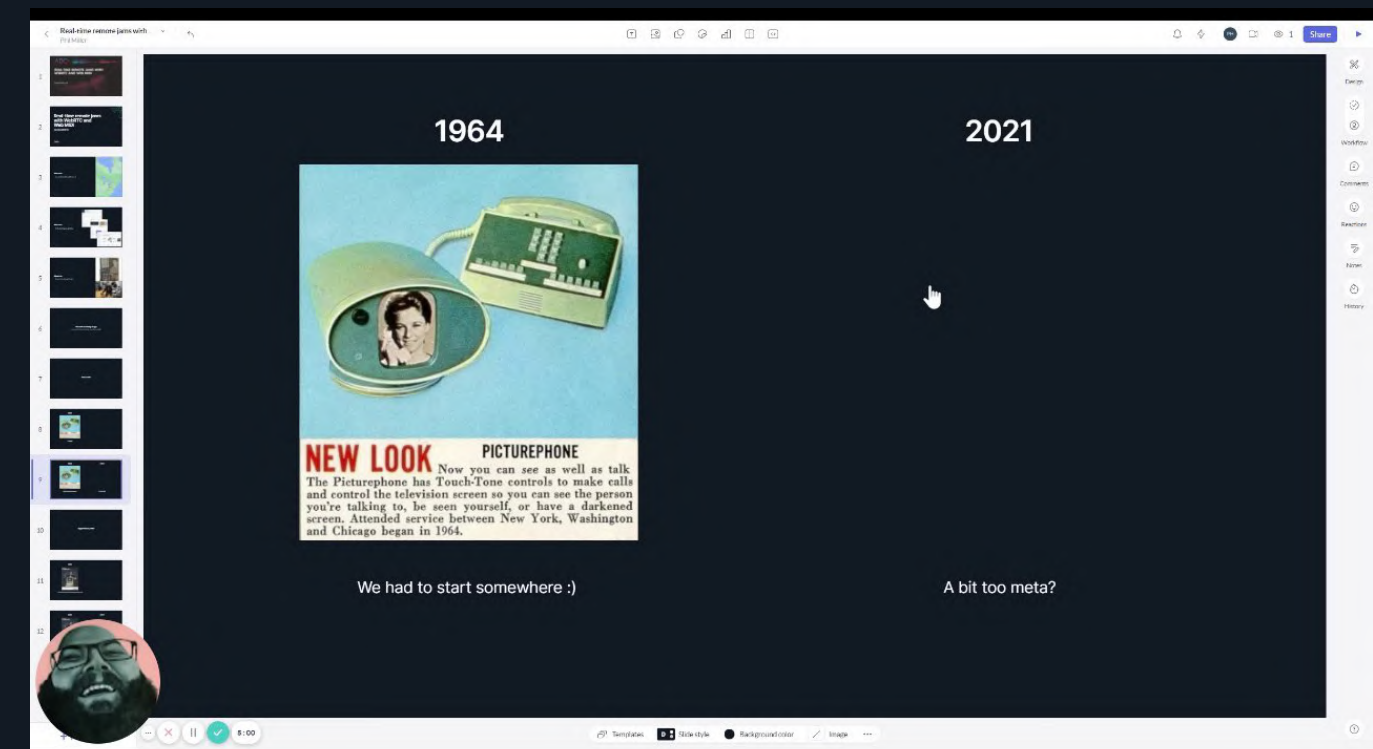


# 1964



We had to start somewhere :)

# 2021



Is this the metaverse?

**Audio / MIDI**


1982\*

NECパーソナルコンピュータ  
PC-8800シリーズ

PC-8801mkII SR

新登場

NEC



すべてを受け継ぎながら、すべてが新しくなった。

本格的なパーソナルユースの時代を実現したベストセラー機・PC-8801mkII。その実績と実力のすべてを受け継ぎながら、飛躍的に成長した後継機がPC-8801mkII SRです。グラフィック・演算処理のスピードをゲームとアップ、512色から8色選べるカラフルなグラフィック機能。パソコンの常識を超えた6重和音の迫力あるサウンド機能など、さらに多彩な機能をプラスして、コストパフォーマンスも抜群。ホビーユースからビジネスまで、バーンデカルな空間をいっしょに拡大してくれる魅力いっぱいのパーソナルコンピュータです。

ベストセラーが超進化した。  
NECパーソナルコンピュータ PC-8800シリーズ  
**PC-8801mkII SR**

model 1811 (1.2MB・5.25インチフロッピーディスク) 標準価格 ¥18,000 (税別) model 2811 (1.2MB・5.25インチフロッピーディスク) 標準価格 ¥18,000 (税別) model 3811 (1.2MB・5.25インチフロッピーディスク) 標準価格 ¥18,000 (税別) model 4811 (1.2MB・5.25インチフロッピーディスク) 標準価格 ¥18,000 (税別)

*\*This model is from 1985 but the ad was too good not to use*




1982\*

NECパーソナルコンピュータ  
PC-8800シリーズ

NEC

PC-8801mkII SR

新登場



すべてを受け継ぎながら、すべてが新しくなった。

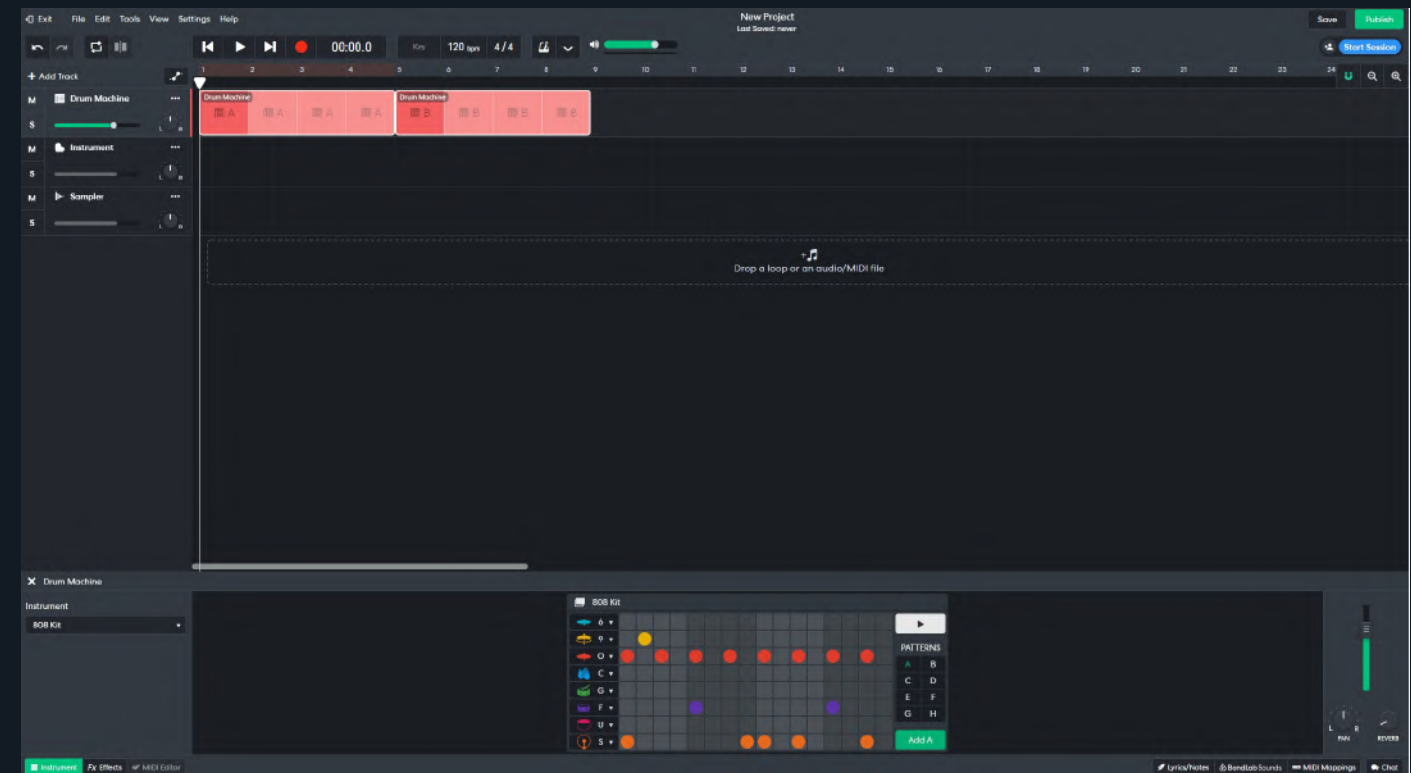
本格的なパーソナルユースの時代を実現したベストセラー機・PC-8801mkII。その実績と実力のすべてを受け継ぎながら、飛躍的に成長した後継機がPC-8801mkII SRです。グラフィック・演算処理のスピードをターボアップ、512色から8色選べるカラフルなグラフィック機能、パソコンの常識を超えた多重和音の迫力あるサウンド機能など、さらに多彩な機能をプラスして、コストパフォーマンスも抜群。ホビーユースからビジネスまで、パーソナルな空間をいっしょに拡大してくれる魅力いっぱいのパーソナルコンピュータです。

ベストセラーが超進化した。  
NECパーソナルコンピュータ PC-8800シリーズ  
**PC-8801mkII SR**

model 1811は720ピクセルディスプレイ、標準価格 108,000円。model 2811は720ピクセルディスプレイ、標準価格 113,000円。model 3811は720ピクセルディスプレイ、標準価格 118,000円。

*\*This model is from 1985 but the ad was too good not to use*

2021



Wait, this is in a browser?

**What if we combined the two?**

**But first:**

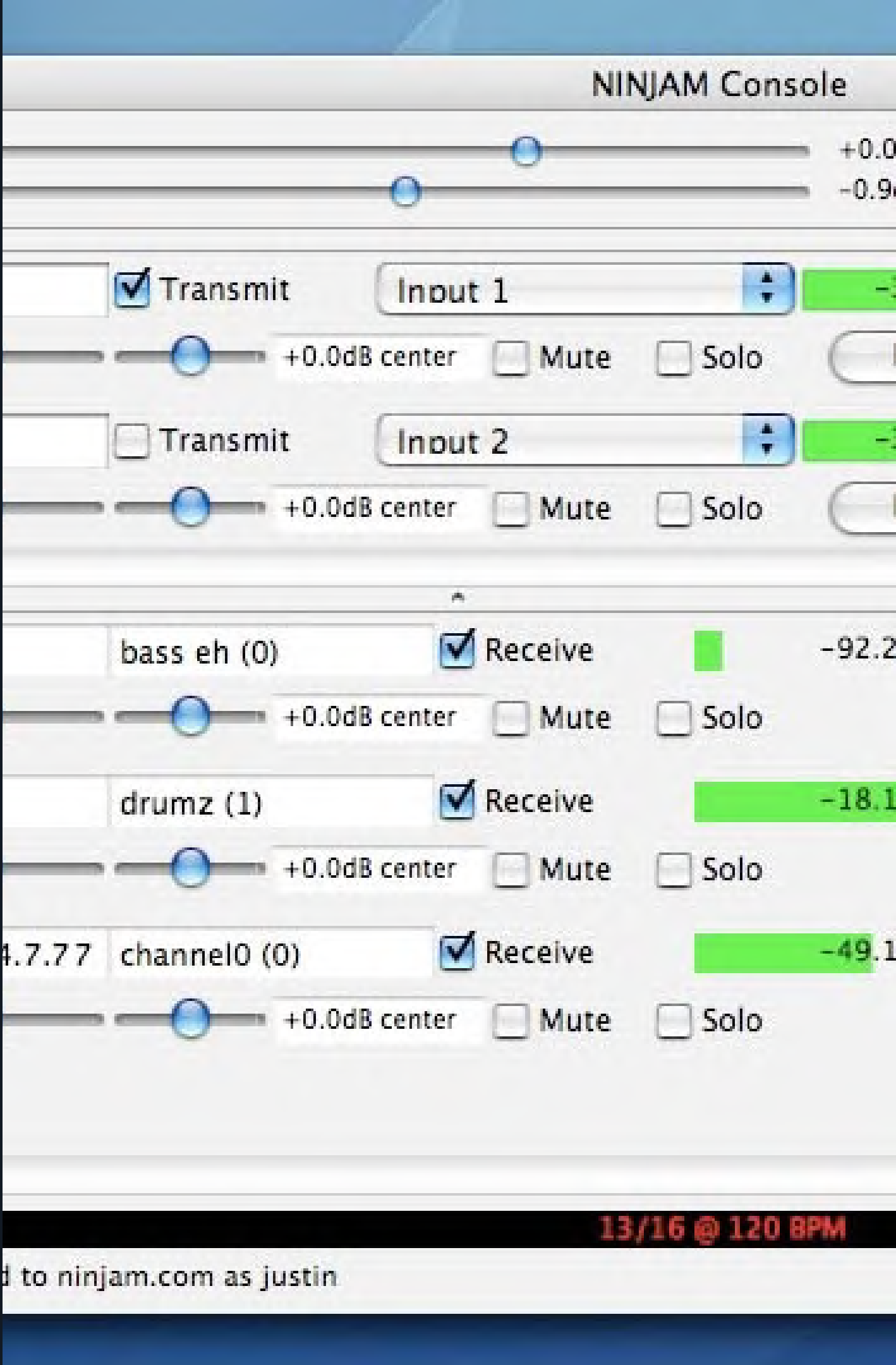
**A brief aside about online music collaboration and latency**



*AKA: you can't change the speed of light*

# Dealing with latency

1. Add it



# Dealing with latency

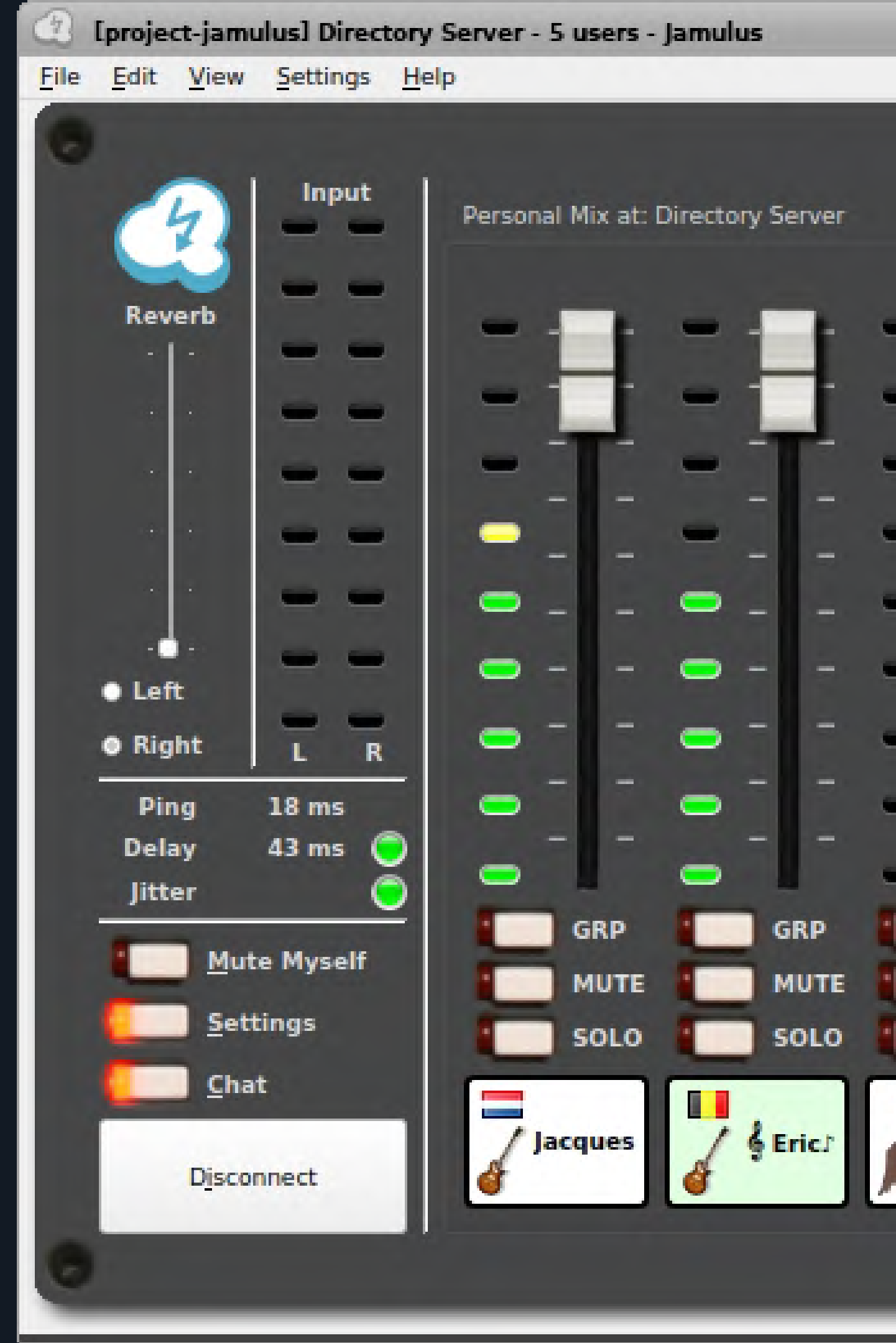
1. Add it
2. Centralize it





# Dealing with latency

1. Add it
2. Centralize it
3. Minimize it

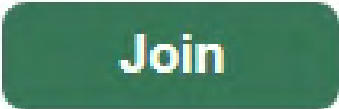


# Dealing with latency

- 1. Add it (and synchronize)
- 2. Centralize it
- 3. Minimize it
- 4. *Embrace it*



## Call controls



## Playback

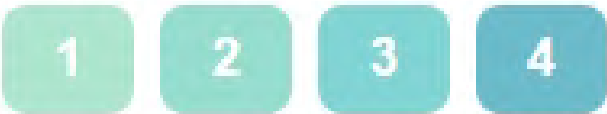


## Digitone Tracks

Mute



Unmute



## Patterns

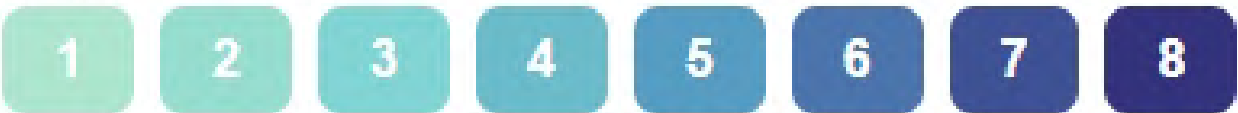


## Digitakt Tracks

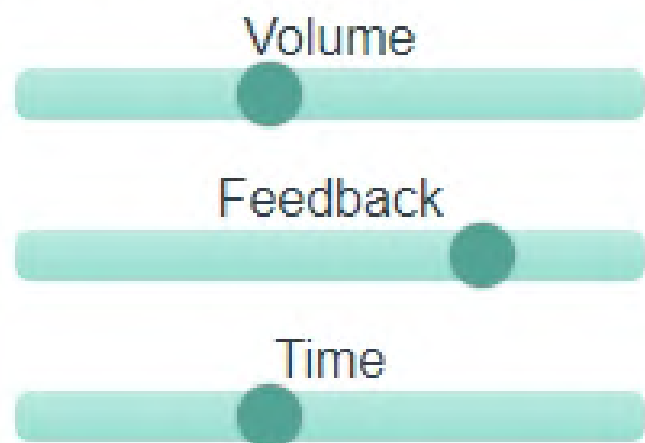
Mute



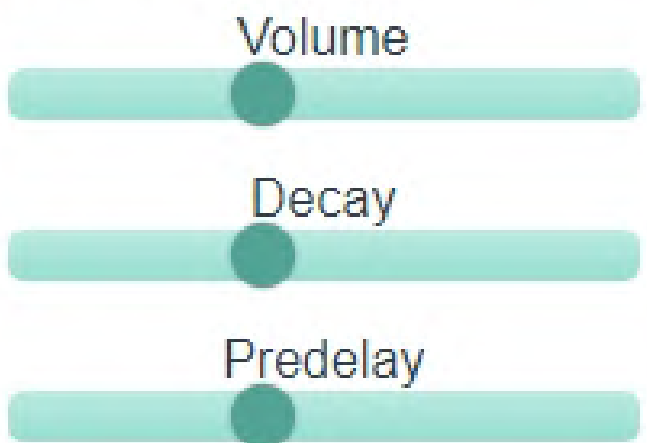
Unmute



## Delay



## Reverb



But how do we "embrace" it?





# Embracing it

1. Have a single audio "host"



# Embracing it

1. Have a single audio "host"
2. Make it primarily loop based





# Embracing it

- 1. Have a single audio "host"
- 2. Make it primarily loop based
- 3. Focus on things that can *feel* immediate, even if they're not
  - Changing phrases/loops (MIDI PC)
  - Changing sound parameters (MIDI CC)



## FX parameters reverb

Parameter name	Description	CC MSB	CC LSB	CC min	CC max	NRPN MSB	NRPN LSB	NRPN min	NRPN max	On
Reverb predelay		21	53	0	127	2	20	0	127	0-
Reverb decay time		74		0	127	2	21	0	127	0-
Reverb shelving frequency		75		0	127	2	22	0	127	0-
Reverb shelving gain		22	54	0	127	2	23	0	127	0-
Reverb highpass filter		76		0	127	2	24	0	127	0-
Reverb lowpass filter		77		0	127	2	25	0	127	0-
Reverb mix volume		23		0	127	2	26	0	127	0-

## FX master

Parameter name	Description	CC MSB	CC LSB	CC min	CC max	NRPN MSB	NRPN LSB	NRPN min	NRPN max	On
Input L volume		24	56	0	127	2	30	0	127	0-
Input R volume		25	57	0	127	2	32	0	127	0-
Pan L		78		0	127	2	3	0	127	0-
Pan R		79		0	127	2	33	0	127	0-
Maxter chorus send		26	58	0	127	2	34	0	127	0-
Master delay send		27	59	0	127	2	35	0	127	0-
Master reverb send		28	60	0	127	2	36	0	127	0-
Master overdrive		29	61	0	127	2	37	0	127	0-
Master pattern		95		0	127	2	38	0	127	0-



# WEBMIDIRTC

or: *How I learned to stop worrying  
and love the browser*

# The Stack - WebRTC

WebRTC is an open-source project to "*enable rich, high-quality RTC applications to be developed for the browser, mobile platforms, and IoT devices, and allow them all to communicate via a common set of protocols*".

- Support in most\* modern browsers
- Peer-to-peer

*\*implementation varies*

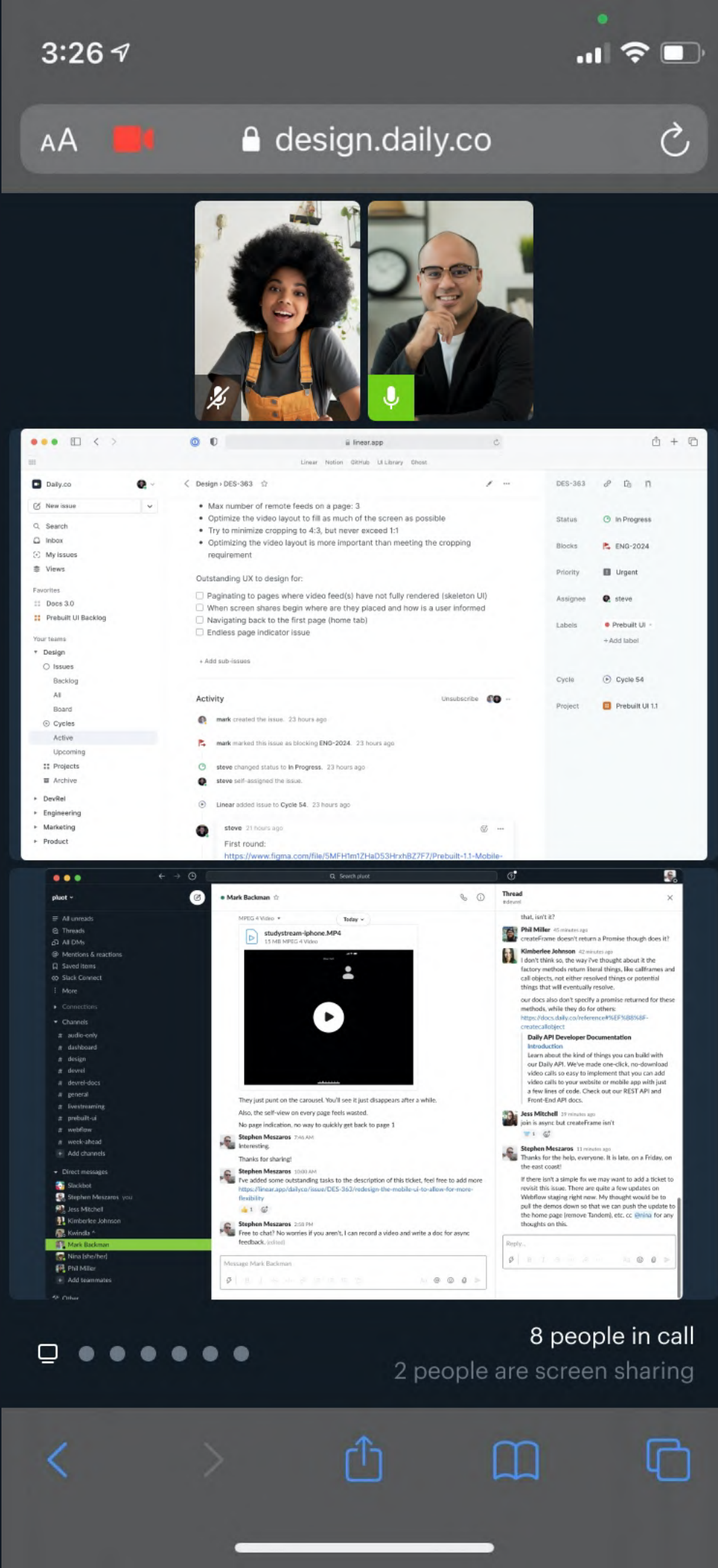
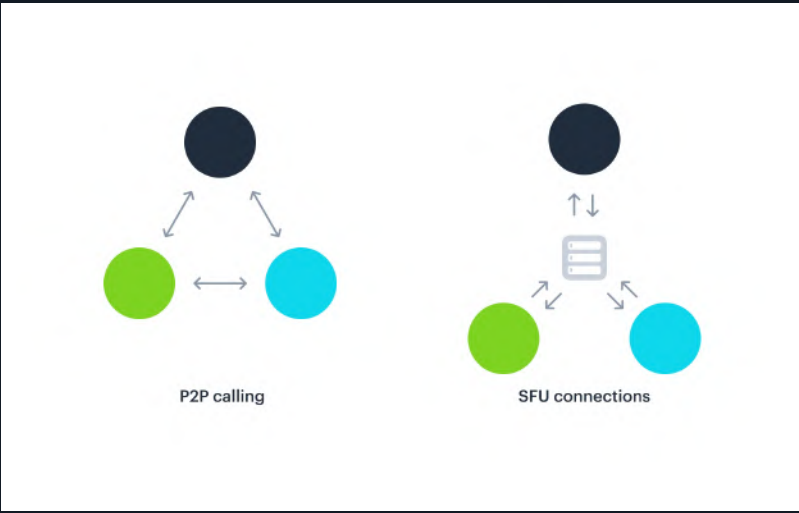
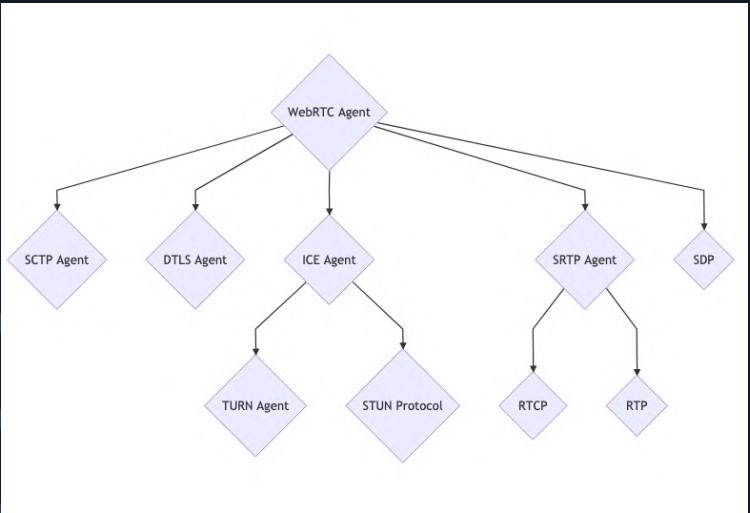




# The Stack - WebRTC (cont'd)

Daily is built on top of WebRTC, with developer-time-to-value, and call quality and reliability as north stars.

- User friendly APIs
- Global infrastructure
  - p2p <> SFU
  - recording
  - live streaming
- Fully featured Prebuilt UI
- World class support
- Call data and analytics



8 people in call

2 people are screen sharing

# The Stack - Front end

Vue is a "progressive [javascript] framework for building user interfaces."

- Incrementally adoptable
- Core library + supporting libraries
- Great standard tooling (vue-cli)
- Single File Components (SFC) are approachable (IMO)

```
<template>
  <main class="page-container">
    <div class="daily-call-container">
      <div class="daily-call" id="call-wrapper"></div>
    </div>
    <div class="controls-container">
      <div class="main-controls">
        <p class="flash-message">
          {{ message || "[See incoming changes here]" }}
        </p>
        <div class="call-controls">
          <div class="daily-controls">
            <h3>Call controls</h3>

            <button class="join-button" @click="joinCall()" :disabled="joined">
              Join
            </button>
          </div>
          <div>
            <h3>Playback</h3>

            <div class="button-container">
              <button class="play-button" @click="controlMessage('play')">
                
              </button>
              <button class="stop-button" @click="controlMessage('stop')">
                
              </button>
            </div>
          </div>
        </div>
      </div>
      <div class="column-section">
        <div>
          <h3>Digitone Tracks</h3>
          <div class="control-section">
            <h4>Mute</h4>

            <div class="button-container">
              <button @click="controlMessage('muteDN', 1)">1</button>
              <button @click="controlMessage('muteDN', 2)">2</button>
              <button @click="controlMessage('muteDN', 3)">3</button>
              <button @click="controlMessage('muteDN', 4)">4</button>
            </div>
          </div>
          <div class="control-section">
            <h4>Unmute</h4>
            <div class="button-container">
              <button @click="controlMessage('unmuteDN', 1)">1</button>
              <button @click="controlMessage('unmuteDN', 2)">2</button>
              <button @click="controlMessage('unmuteDN', 3)">3</button>
              <button @click="controlMessage('unmuteDN', 4)">4</button>
            </div>
          </div>
        </div>
      </div>
    </div>
  </main>
</template>
```



# The Stack - Front end (cont'd)

Web MIDI allows a browser to interact with MIDI devices via a standard set of APIs

- Supported in Chrome, Opera, Android WebView, Edge
- Possible in other browsers via Jazz-Plugin
- WebMidi.js
  - provides more convenient higher level APIs
  - much better DX

```
// Enable WebMidi.js
WebMidi.enable(function (err) {

  if (err) {
    console.log("WebMidi could not be enabled.", err);
  }

  // Viewing available inputs and outputs
  console.log(WebMidi.inputs);
  console.log(WebMidi.outputs);

  // Reacting when a new device becomes available
  WebMidi.addListener("connected", function(e) {
    console.log(e);
  });

  // Reacting when a device becomes unavailable
  WebMidi.addListener("disconnected", function(e) {
    console.log(e);
  });

  // Display the current time
  console.log(WebMidi.time);

  // Retrieving an output port/device using its id, name or index
  var output = WebMidi.getOutputById("123456789");
  output = WebMidi.getOutputByName("Axiom Pro 25 Ext Out");
  output = WebMidi.outputs[0];

  // Play a note on all channels of the selected output
  output.playNote("C3");

  // Play a note on channel 3
  output.playNote("Gb4", 3);

  // Play a chord on all available channels
  output.playNote(["C3", "D#3", "G3"]);

  // Play a chord on channel 7
  output.playNote(["C3", "D#3", "G3"], 7);

  // Play a note at full velocity on all channels)
  output.playNote("F#-1", "all", {velocity: 1});

  // Play a note on channel 16 in 2 seconds (relative time)
  output.playNote("F5", 16, {time: "+2000"});

  // Play a note on channel 1 at an absolute time in the future
  output.playNote("F5", 16, {time: WebMidi.time + 3000});
```

# The Stack - Hardware

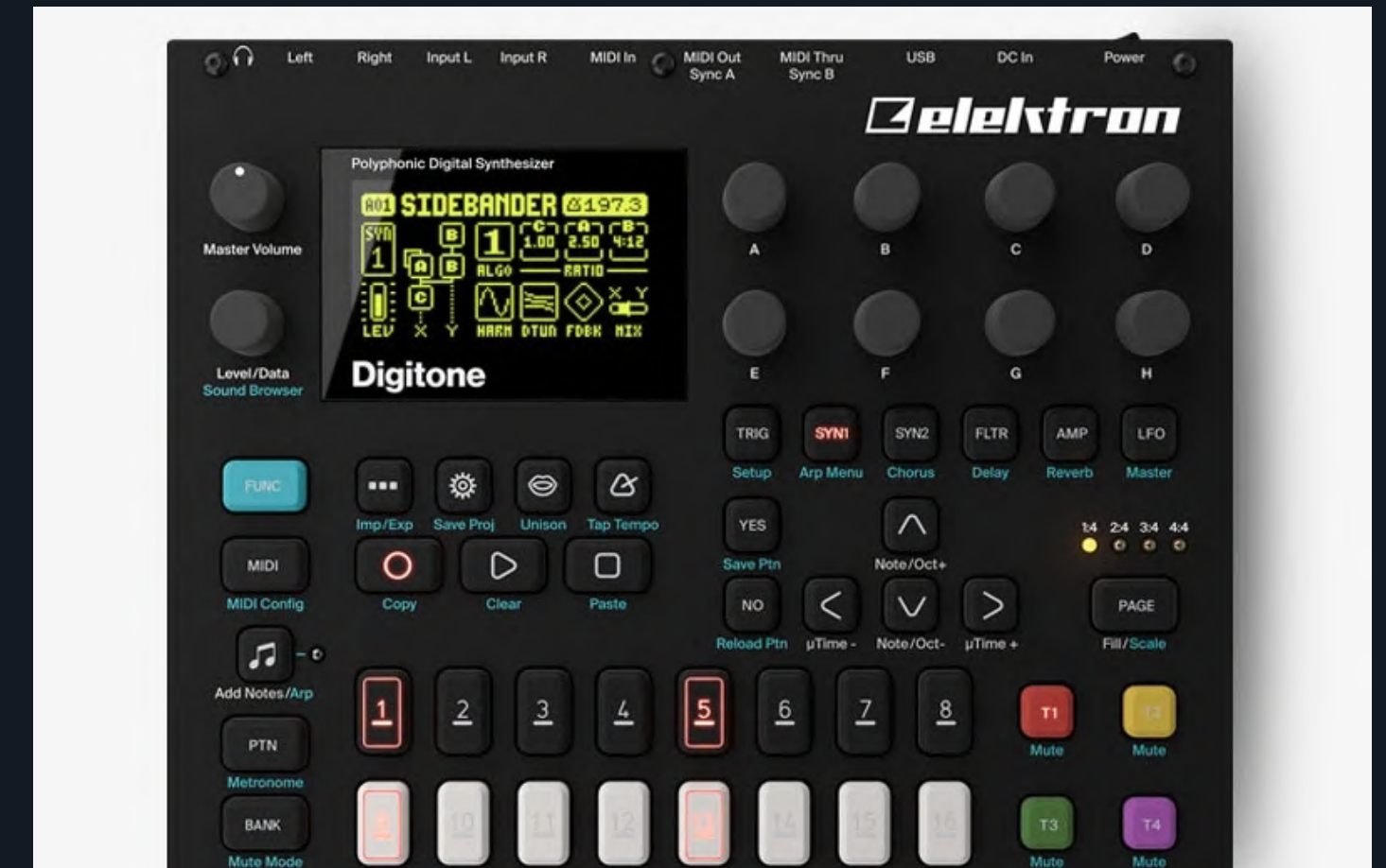
## Digitakt

- Drum computer ( 8 tracks)
- Sampler
- MIDI Sequencer (8 tracks)



## Digitone

- "Modern" FM synth
- 8 Voices (4 tracks)
- MIDI Sequencer (4 tracks)



# A quick tour of the app

**Now let's look at the code**



DEMO TIME 🙌

**So what next?**

# What next?

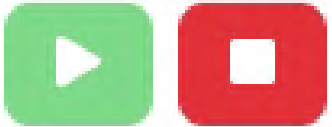
1. Add more PC/CC
2. More devices
3. Remote MIDI
4. Remote sampling/looping
5. More participant roles
6. Command queuing
7. Recording, Live streaming



## Call controls

Join

## Playback



## Digitone Tracks

Mute



Unmute



## Patterns

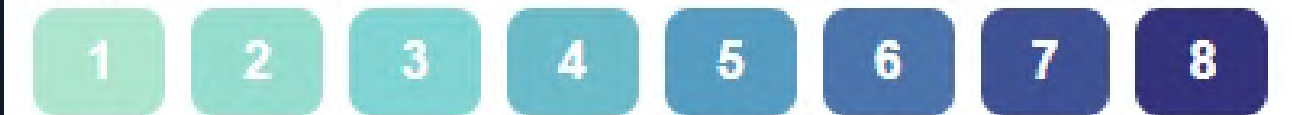


## Digitakt Tracks

Mute



Unmute



## Delay

Volume



Feedback



Time



## Reverb

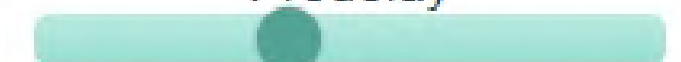
Volume



Decay



Predelay



# Practical applications

1. Interactive live sessions/streams
2. Remote production
3. Music retail
4. Product launches / previews





Questions?

## Shout out to:

- All the OSS folks (Vue, WebMidi.js)
- Daily
- ADC21 Team





ADC<sup>21</sup>



# REAL-TIME REMOTE JAMS WITH WEBRTC AND WEB MIDI

PHILIP MILLER

