MAKE MORE THAN MUSIC

WITH TINY COMPUTERS, JAVASCRIPT & MIDI

GEORGE MANDIS

HOLYJS 2017 — ST. PETERSBURG

GEORGE MANDIS

- WEB — george.mand.is
- EMAIL — george@mand.is
- TWITTER — @georgeMandis
- GITHUB — @georgemandis
- WORK — snaptortoise.com

GEORGE MANDIS

FUN FACTS!

- First time in Russia
- Lived as digital nomad in 18 countries
- Born in Saudi Arabia! Long story
- Once unintentionally cheated running a marathon in North Korea
- Most known project — Konami-JS

GEORGE MANDIS

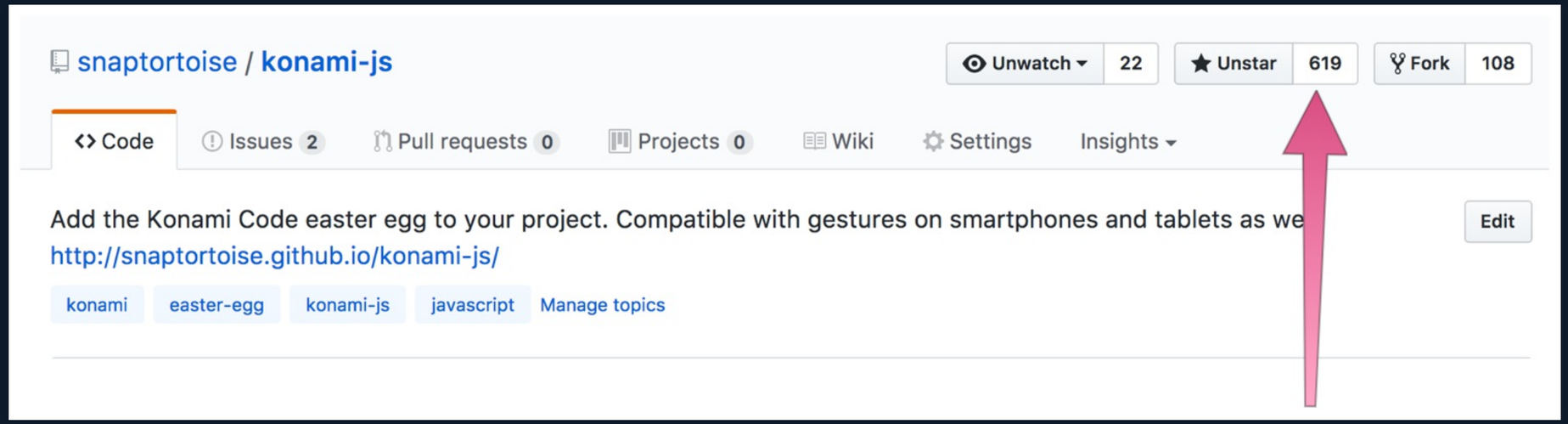
About Konami JS

My history with creating frivolous things.

GEORGE MANDIS

About Konami JS

My history with creating frivolous things.



LET'S START WITH AN EXPERIMENT...

WEBRTC SYMPHONY I

THE INTERACTIVE PIANO RECITAL

PERFORMED (POORLY) BY GEORGE MANDIS

HTTP://HOLYJS.MAND.IS/WS1


WEBRTC SYMPHONY II
JESU, JOY OF MAN'S DESIRING
COURTESY OF BACH
HTTP://HOLYJS.MAND.IS/WS2

WHAT WE'RE TALKING ABOUT

JAVASCRIPT

(Obviously)

TINY COMPUTERS

Arduinos, Rasperry Pis, Espruinos & more

MIDI

The **Musical Instrument Device Interface**


WHY JAVASCRIPT?

JAVASCRIPT IS EVERYWHERE

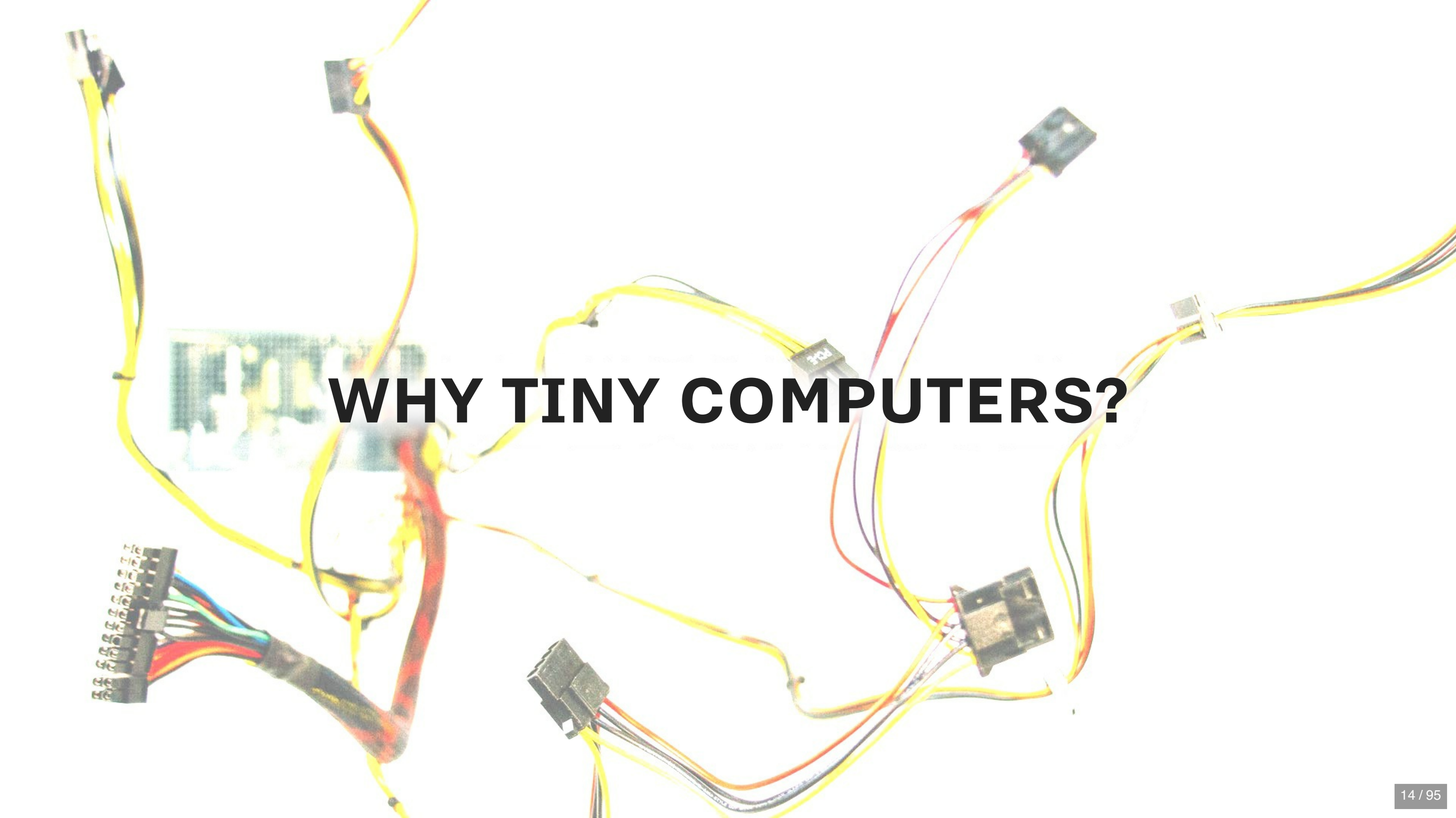
Clients, servers, websites, software, apps, bots, AIs, IoT, embedded tech... It's become the Lingua franca of the web.

IT'S ASYNCHRONOUS NATURE

A natural fit with MIDI

IT'S WHERE THE PEOPLE ARE

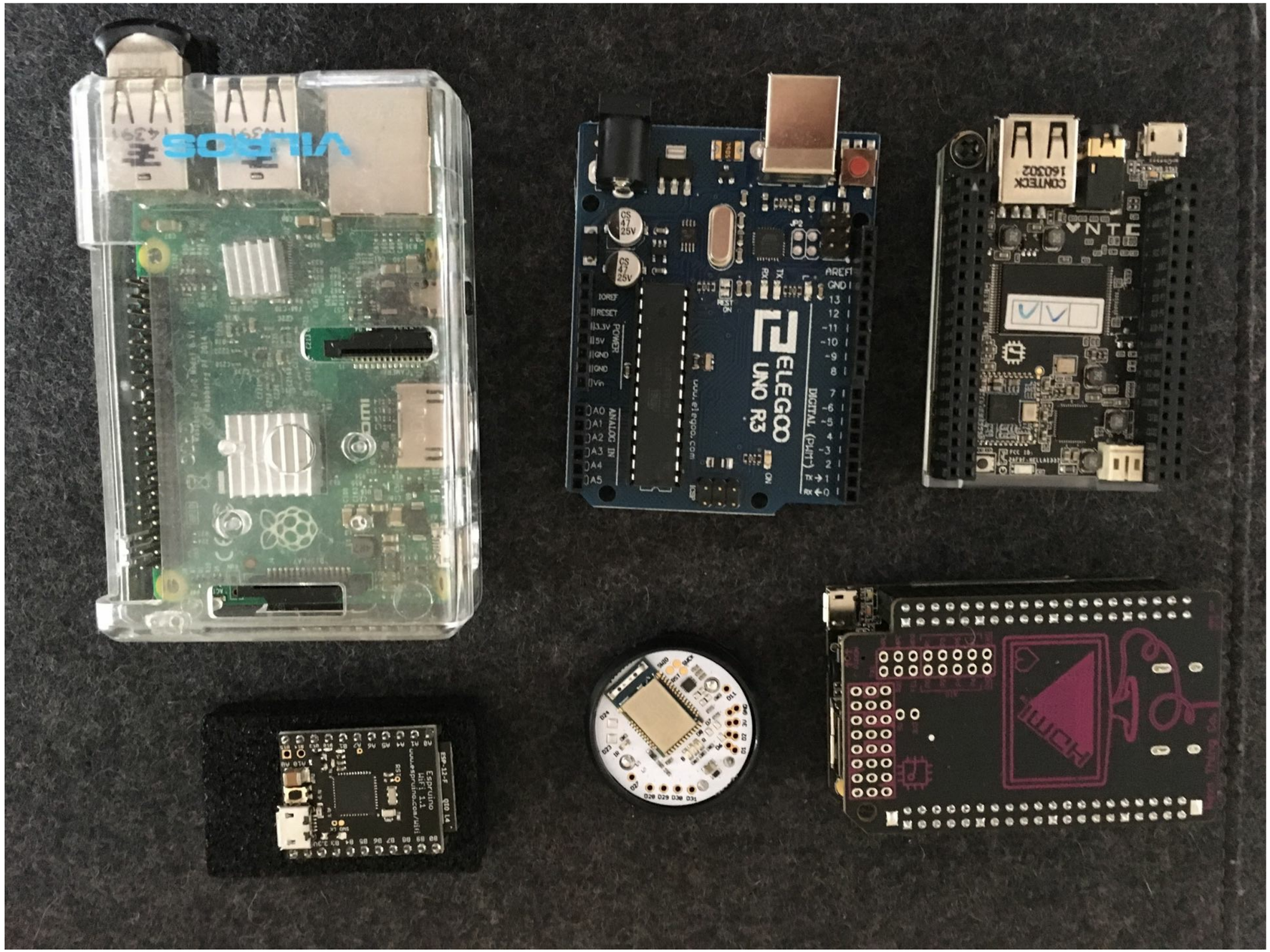
In ~8 years Node.JS has made JavaScript a legitimate server-side contender and fundamentally changed how we develop for the web and beyond.

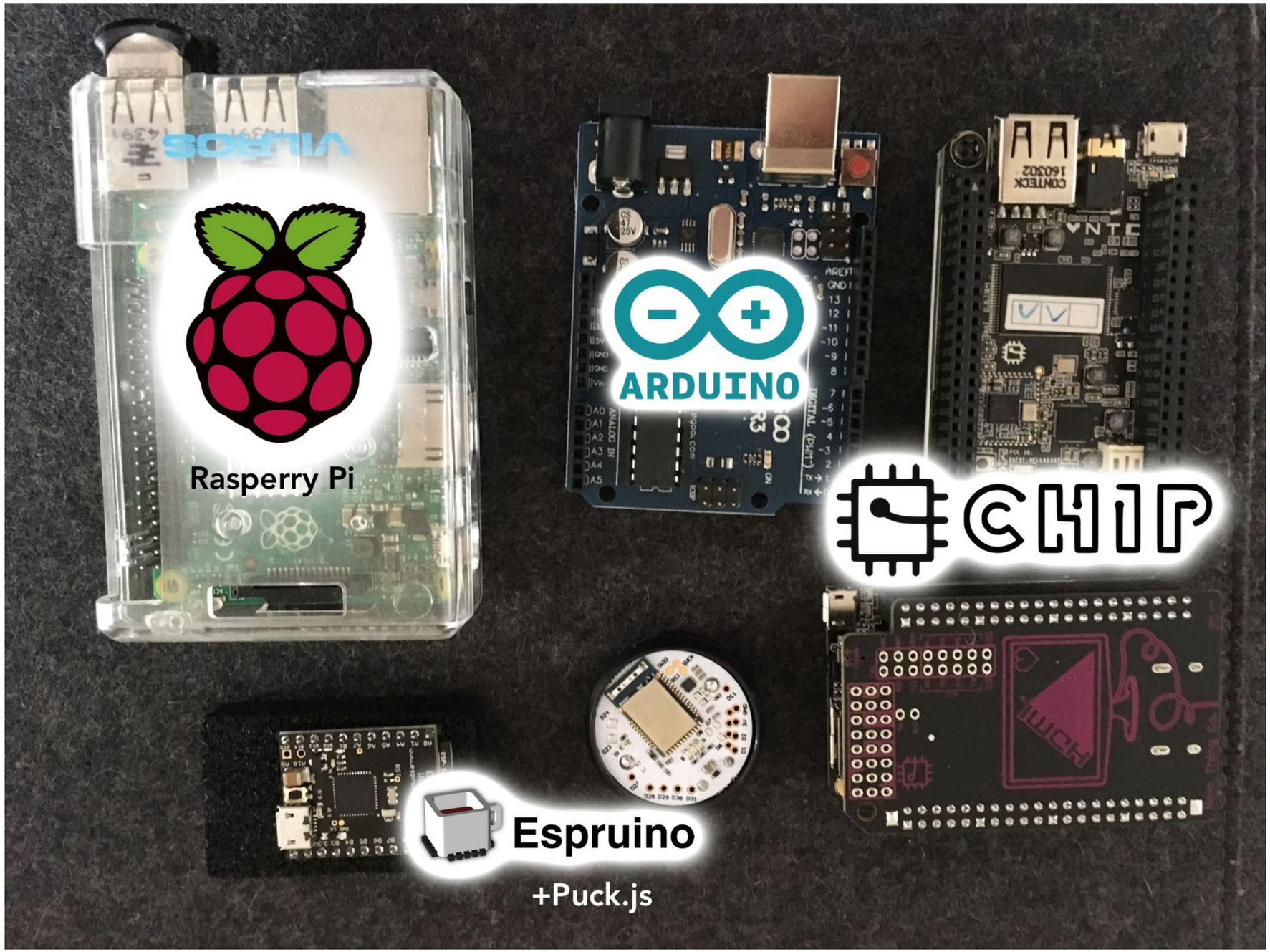


WHY TINY COMPUTERS?

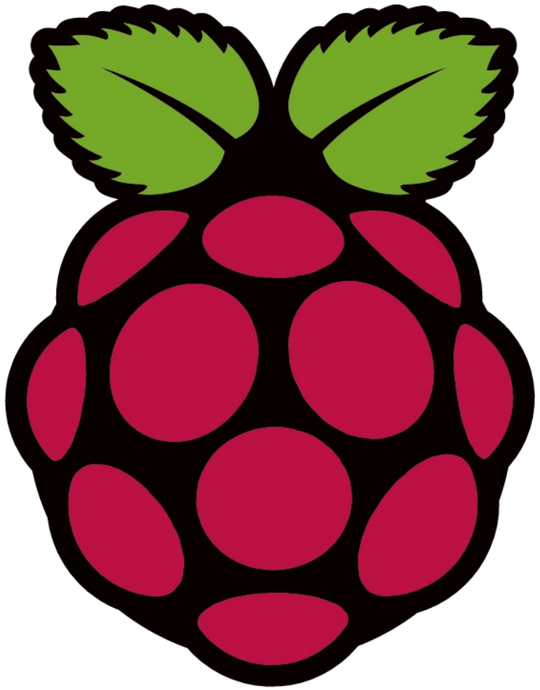
~~WHY TINY COMPUTERS?~~

What *are* Tiny Computers exactly?

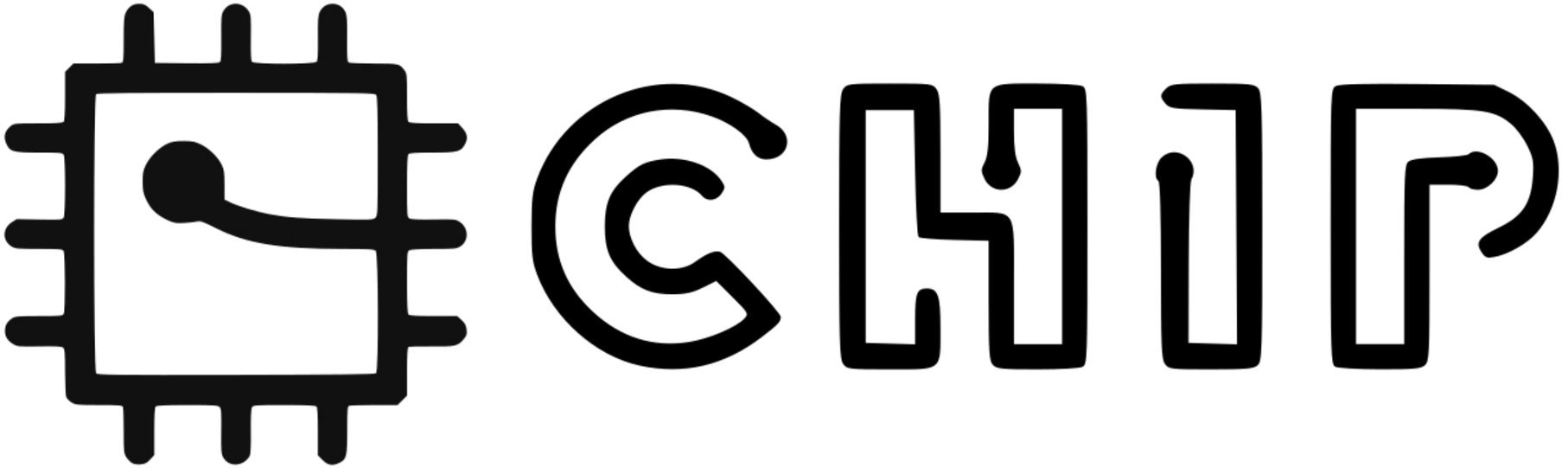

ELEGOO
UNO R3
www.elegoo.com
Espruino
WiFi 1.1

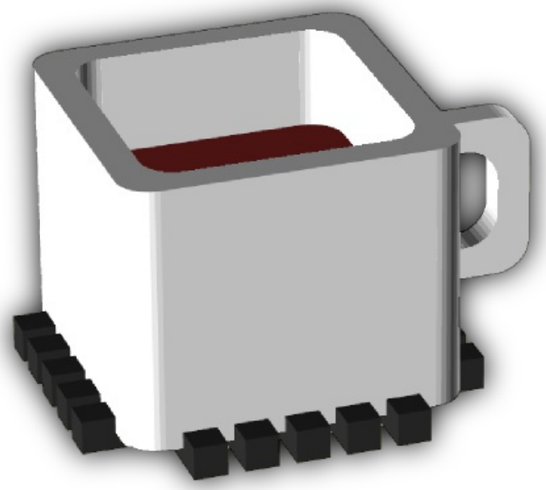

Rasperry Pi
ARDUINO
CHIP
Espruino
+Puck.js


ARDUINO



RASPERRY PI


CHIP



Espruino

They're fast, affordable and easy to use

Great for Stand-Alone Projects and Creative Tinkering

THEY'RE FUN!


NOISE
-1.2 dB
8 ms
FILTER DECAY
15
FILTER VELOCITY
1.51
Analog
Jun-O 106 B
ILTER FREQ
447
Rim 909
Kick 909
2-Kit-909 Cl
1-Jun-O 106 B

WHY MIDI?

~~WHY MIDI?~~

Wait — what is MIDI exactly?

MIDI

Musical **I**nstrument **D**evice **I**nterface

- It's a protocol
- It's a standard
- It's also a file format
- Created in 1983
- Maintained by the MIDI Association since 1985.

IT'S NOT THIS...

“Passport”

by Passport Designs

...OR THIS...

“Canyon”

by George Stone of Passport Designs


...OR EVEN THIS
“Clouds”
by Brian Orr


GENERAL
MIDI

MIDI THE PROTOCOL

- Allowed electronic musical instruments to communicate one another
- Described common instrument performance parameters — *noteOn*, *noteOff*, *pitchbend*, *programChange* and others

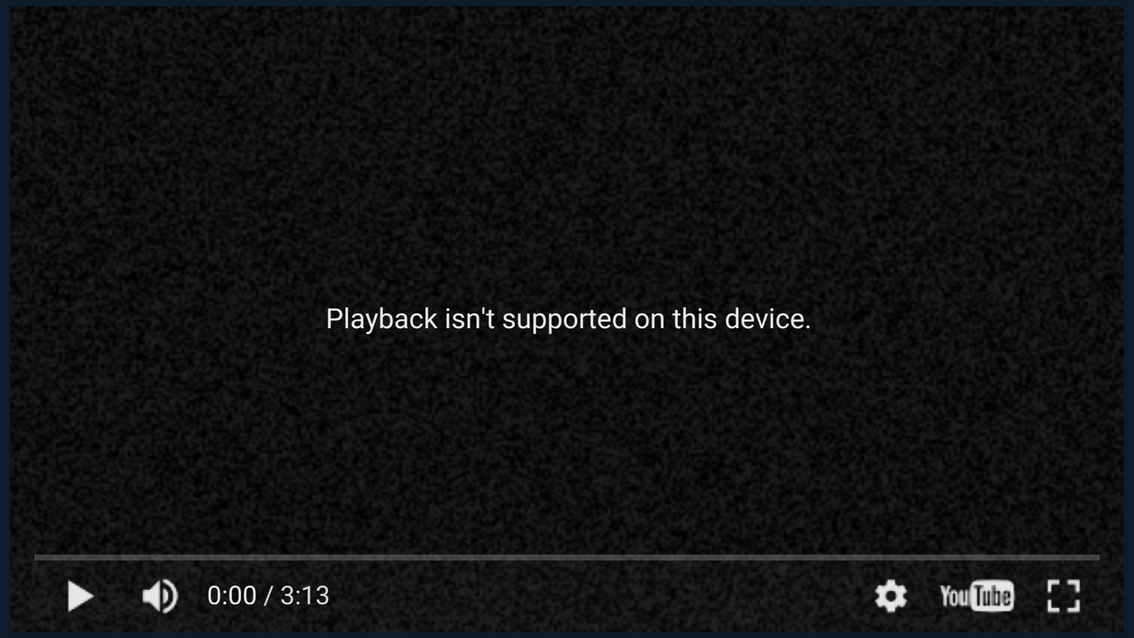
MIDI THE STANDARD

- Roland
- Oberheim
- Sequential Circuits
- Yamaha
- Korg
- Kawai

IKUTARO KAKEHASHI

MIDI THE FILE FORMAT

- Describe MIDI Events
- Describe the timing surrounding those events


Playback isn't supported on this device.
0:00 / 3:13
YouTube

EVEN MORE MIDI

- MIDI Show Control (stage lighting)
- General MIDI 2
- Black MIDI
- MIDI Light Control
- It...
- Never...
- Ends...


BACK TO THE QUESTION...
WHY MIDI?

MIDI IS ASYNCHRONOUS

MIDI TALKS TO HARDWARE

MIDI IS EASY TO REPURPOSE

(I seriuosly wish I could take credit for this)


CHROME SPEAKS MIDI!
:-D
And so do **lots** of other JavaScript projects

MIDI CONTROLLERS

- Pre-existing, USB-ready (plug-and-play)
- Old hardware can be used with a MIDI-USB adapter
- You can build your own!

KEYBOARDS & CONTROL PADS

KEYBOARDS & CONTROL PADS

BUTTONS AND SWITCHES

LESS COMMON MIDI CONTROLLERS

- Expression pedals
- Breathe & bite controllers
- Accelerometer based controllers
- Tenori On
- Pianocade
- Karlax from Da Fact
- Bananas... (via Makey Makey)

BUILD YOUR OWN!

It's pretty easy to build a MIDI in/out device on Arduino.

ARDUINO MIDI CONTROLLER

ANATOMY OF A MIDI MESSAGE

- Every message is 3 bytes.
- There are only 2 types of messages: status and data.
- A status byte always begins with 1 and data bytes always begin with 0
- That leaves 7 bits left per byte for expressing the message
- For a status message, 3 bits of the first byte describe the type of status message; the remaining 4 describe the channel

ANATOMY OF A MIDI MESSAGE

Example:

128x0001 | 0x1111111 | 0x0000000

status: "noteOn", channel: 1, data1: 127, data2: 0

[1, 127, 0]

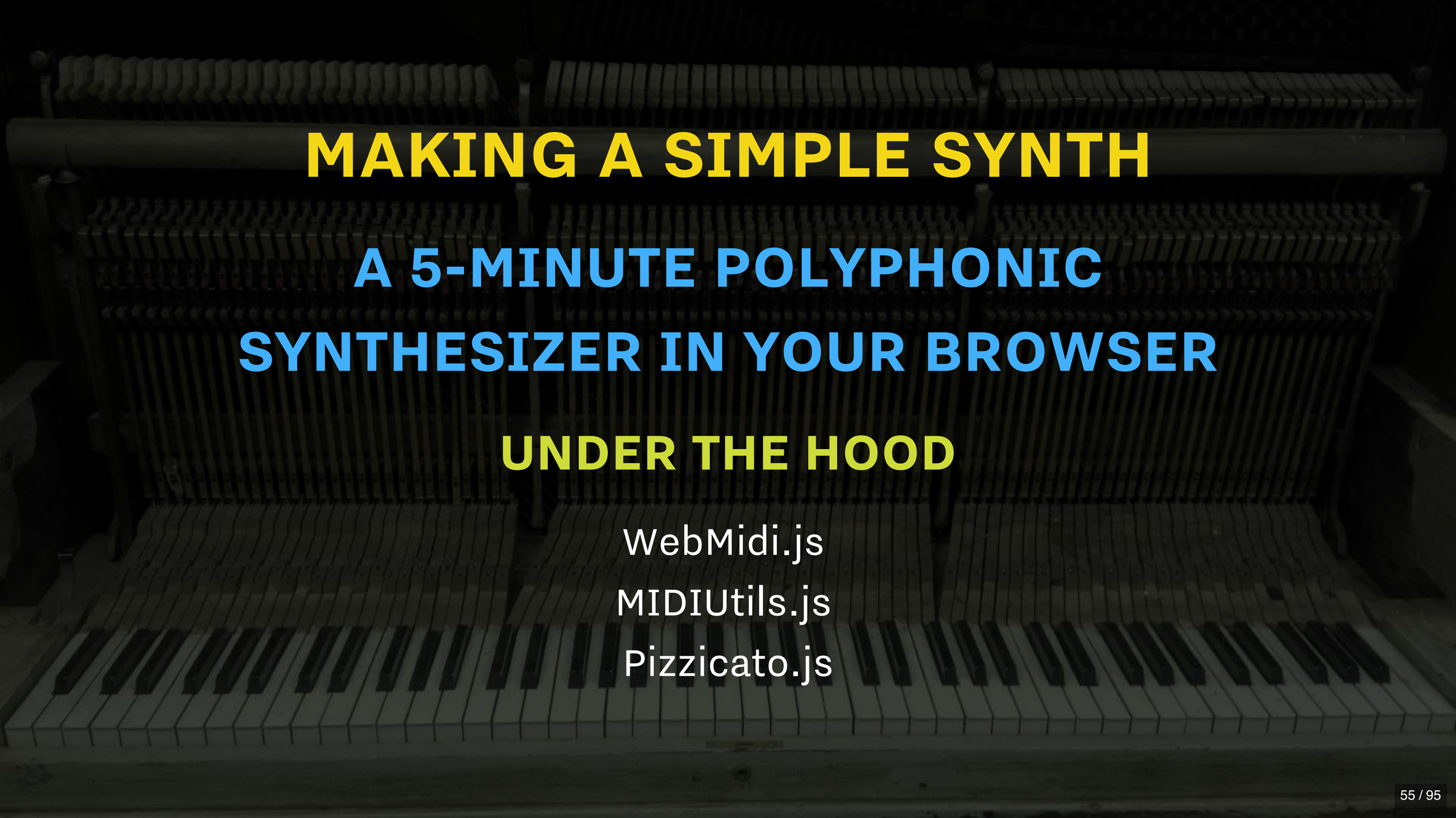
ANATOMY OF A MIDI MESSAGE

Command Name	Status Byte	Data Byte 1	Data Byte 2
Note Off	128 + Channel	0-127 Pitch	0-127 Velocity
Note On	144 + Channel	0-127 Pitch	0-127 Velocity
Poly Pressure	160 + Channel	0-127 Pitch	0-127 Pressure
Control Change	176 + Channel	0 Undefined	0-127 MSB
Control Change	176 + Channel	1 Modulation Wheel	0-127 MSB
Control Change	176 + Channel	2 Breath Controller	0-127 MSB
Control Change	176 + Channel	3 After Touch	0-127 MSB
Control Change	176 + Channel	4 Foot Controller	0-127 MSB
Control Change	176 + Channel	5 Portamento Time	0-127 MSB
Control Change	176 + Channel	6 Data Entry	0-127 MSB
Control Change	176 + Channel	7 Main Volume	0-127 MSB
Control Change	176 + Channel	8-31 Undefined	0-127 MSB
Control Change	176 + Channel	32-63 LSB of 0-31	0-127 MSB
Control Change	176 + Channel	64 Damper Pedal	0:off 127:on
Control Change	176 + Channel	65 Portamento	0:off 127:on
Control Change	176 + Channel	66 Sostenuto	0:off 127:on
Control Change	176 + Channel	67 Soft Pedal	0:off 127:on
Control Change	176 + Channel	68-92 Undefined	0:off 127:on
Control Change	176 + Channel	93 Chorus	0:off 127:on
Control Change	176 + Channel	94 Celeste	0:off 127:on
Control Change	176 + Channel	95 Phaser	0:off 127:on

ANATOMY OF A MIDI FILE

- Broken down into 3-sectioned chunks: type, length and data
- Two types of chunks: header and track

BEFORE WE GET TOO FAR INTO “MORE” THAN MUSIC WITH MIDI...



MAKING A SIMPLE SYNTH

A 5-MINUTE POLYPHONIC SYNTHESIZER IN YOUR BROWSER

UNDER THE HOOD

WebMidi.js

MIDIUtils.js

Pizzicato.js

WEB MIDI & WEB AUDIO

FUN WITH CONTROL CHANGES

UNDER THE HOOD

WebMidi.js

MIDIUtils.js

Pizzicato.js

WEBRTC SYMPHONY I

THE INTERACTIVE PIANO RECITAL

UNDER THE HOOD

Peer.js

MIDIUtils.js

Pizzicato.js

WEBRTC SYMPHONY II

SHARING PARTS OF MIDI FILE

UNDER THE HOOD CLIENT-SIDE

Express

Socket.io

MIDIUtils.js

midifile

midievents

OKAY, NOW LET'S LOOK AT SOME OF THE “MORE” THAN MUSIC WITH MIDI...

MIDI COLOR MIXER

BUILD AN RGB AND HSL MIXER/PALETTE

UNDER THE HOOD

WebMidi.js

TinyColor.js


MIDI & CSS FILTERS
MANIPULATE BLEEDING-EDGE CSS FEATURES
UNDER THE HOOD
WebMidi.js

MIDI PIXEL ART

DRAW SIMPLE IMAGES ON A LAUNCHPAD

UNDER THE HOOD

Native WebMIDI implementation!

```
var x = noteNumber & 0x0f;
var y = (noteNumber & 0xf0) >> 4;
```

MIDI HOT HAND COLOR MIXER

ANOTHER COLOR MIXER WITH A MORE ABSTRACT CONTROLLER

UNDER THE HOOD

WebMidi.js

MIDIUtils.js

MIDI HOT HAND COLOR CONTROL GAME

A TWIST ON THE COLOR MIXER, AS A GAME

UNDER THE HOOD

WebMidi.js

MIDIUtils.js

MIDI WHACK-A-MOLE GAMME

RECREATING A SILLY ARCADE GAME ON THE LPD8

UNDER THE HOOD

WebMidi.js

MIDI GO

TURNING A NOVATION LAUNCHPAD INTO THE CLASSIC STRATEGY GAME

UNDER THE HOOD

Tenuki.js

(AlphaGo AI not included)

THIS SLIDE DECK!

THE SLIDE DECK

OVERVIEW

Frameworks

Reveal.js

WebMidi.js

Controllers

Puck.js

Logidy UMI3

Akai LPD8

THE SLIDE DECK

GOALS

Use MIDI controllers to...

- Go forward and backwards through the slide deck
- Trigger “interactions” in active slide for media
- Switch program modes to alter these controller behaviors

THE SLIDE DECK

CHALLENGES

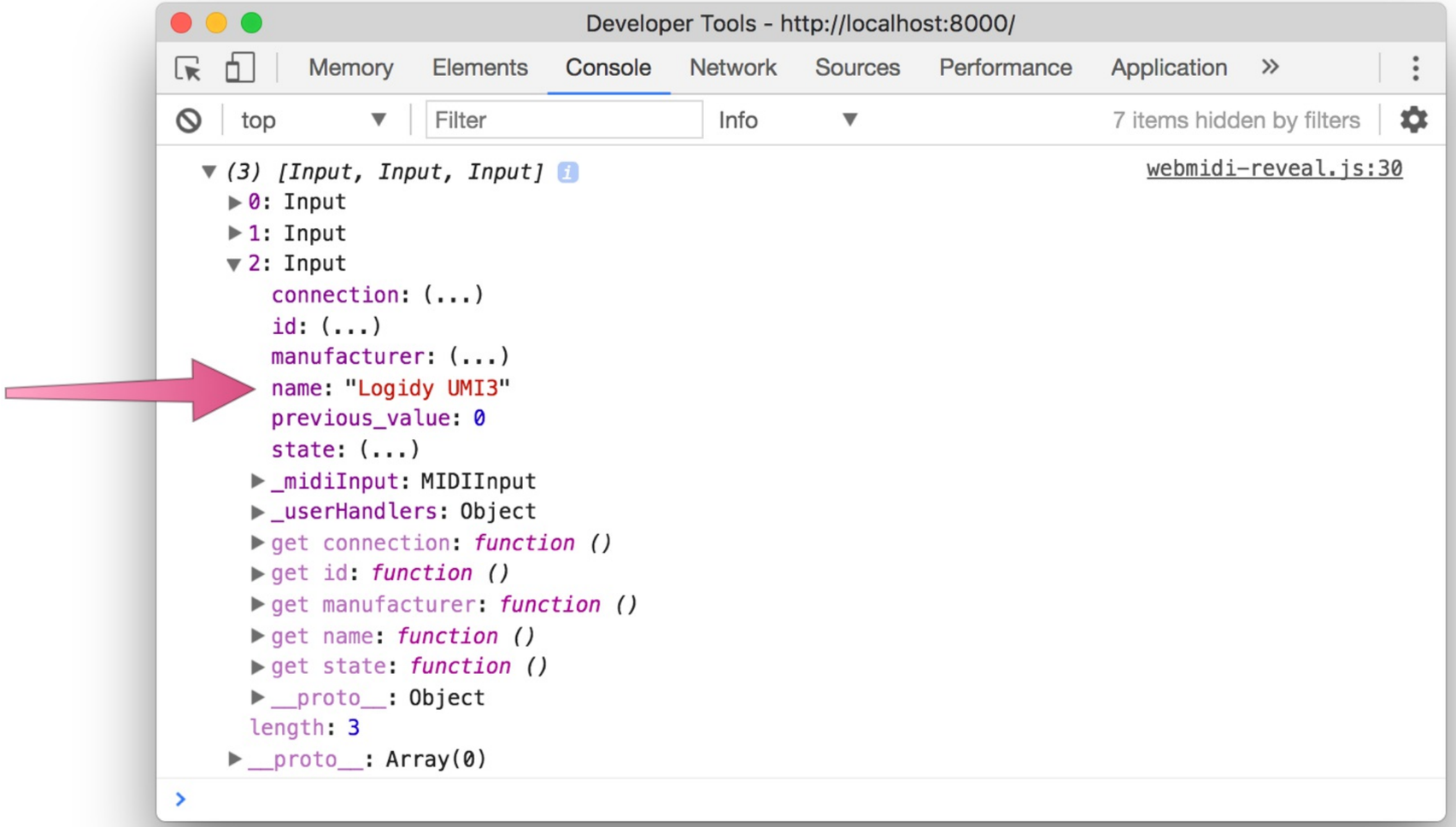
- Recognize devices regardless of port
- Map MIDI messages to Reveal.js API
- Store and trigger program changes

THE SLIDE DECK

REFERENCING DEVICES BY NUMBER

The default **WebMidi.inputs** array isn't ideal.

```
</>
console.log(WebMidi.inputs);
```


Developer Tools - http://localhost:8000/
Memory
Elements
Console
Network
Sources
Performance
Application
top
Filter
Info
7 items hidden by filters
webmidi-reveal.js:30
(3) [Input, Input, Input]
0: Input
1: Input
2: Input
connection: (...)
id: (...)
manufacturer: (...)
name: "Logidy UMI3"
previous_value: 0
state: (...)
_midiInput: MIDIInput
_userHandlers: Object
get connection: function ()
get id: function ()
get manufacturer: function ()
get name: function ()
get state: function ()
__proto__: Object
length: 3
__proto__: Array(0)

THE SLIDE DECK

REFERENCING DEVICES BY NUMBER

If I put this down for a couple weeks I will probably forget which controller **WebMidi.inputs[0]** was supposed to be.

```
</>
if (WebMidi.inputs[0]) {
  WebMidi.inputs[0].addListener('noteon', '1', function(e){
    if (e.data[0] == 144 && e.data[2] == 64 ) {
      var value = e.data[1];
      if (value === 60) Reveal.left();
      if (value === 64) Reveal.right()
      if (value === 62) triggerMedia();
    }
  });
```

THE SLIDE DECK

REFERENCING DEVICES BY NAME

The **getInputByName** method helps clarify what device we're talking about.

```
WebMidi.getInputByName['LPD8']
```

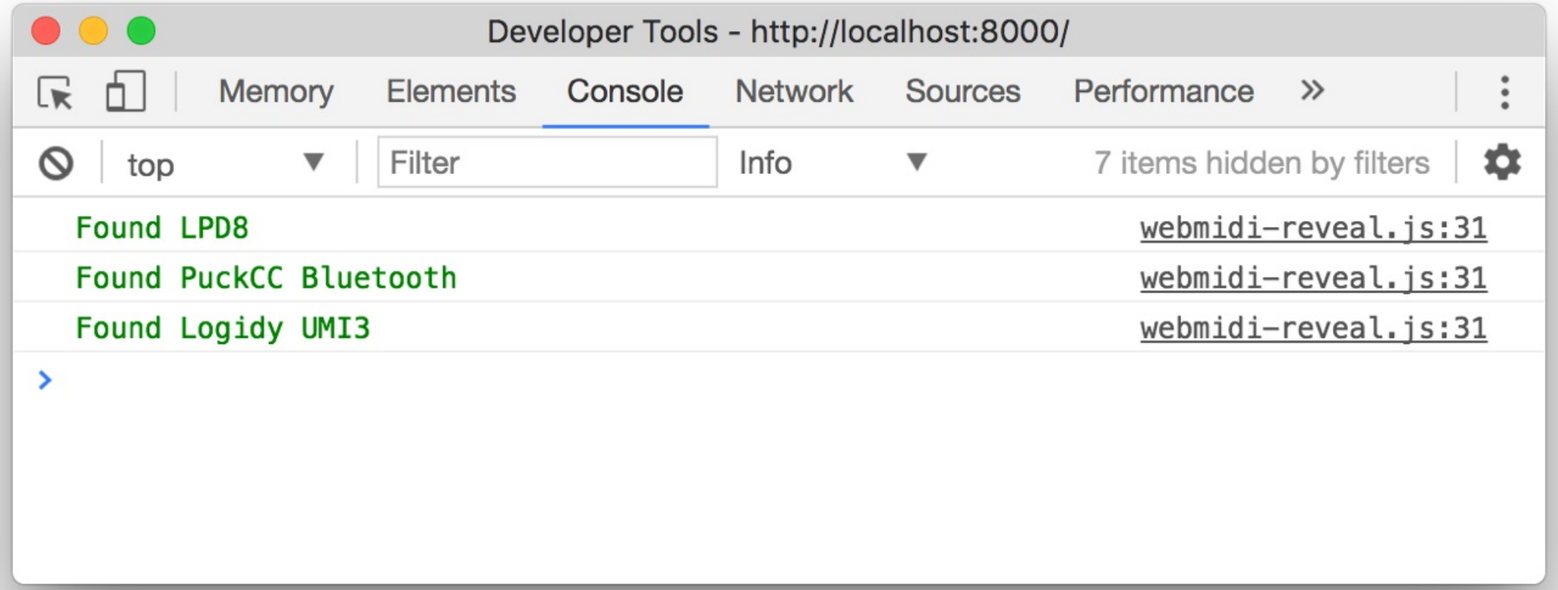
THE SLIDE DECK

REFERENCING DEVICES BY NAME

We can make this a little nicer.

```
</>
var midiInputs = {};

WebMidi.inputs.forEach(function(input) {
  console.log("%cFound " + input.name, "color:green");
  midiInputs[input.name] = input;
});
```


Developer Tools - http://localhost:8000/
Memory
Elements
Console
Network
Sources
Performance
top
Filter
Info
7 items hidden by filters
Found LPD8
webmidi-reveal.js:31
Found PuckCC Bluetooth
webmidi-reveal.js:31
Found Logidy UMI3
webmidi-reveal.js:31

THE SLIDE DECK

REFERENCING DEVICES BY NAME

In practice this is easier when juggling multiple devices.

```
</>
if (midiInputs['Logidy UMI3']) {
  midiInputs['Logidy UMI3'].addListener('noteon', '1', function(e) {
    if (e.data[0] == 144 && e.data[2] == 64 ) {
      var value = e.data[1];
      if (value === 60) Reveal.left();
      if (value === 64) Reveal.right()
      if (value === 62) triggerMedia();
    }
  });
```

THE SLIDE DECK

MAPPING MESSAGES TO REVEAL.JS

- Learn each controller on a case-by-case basis
- Figure out what makes sense for that device
- Sometimes you can change the hardware behavior

THE SLIDE DECK

MAPPING MESSAGES TO REVEAL.JS

Using the Logidy UMI3, Akai LPDB8 and Puck.js provides:

- 3 on/off buttons on the floor
- 1 expression pedal on the floor
- 8 pads + 8 knobs on desk
- 1 program change signal via button
- 1 Wirelss button with 3 types of control-changes


UMI3
Parametric USB MIDI Foot Controller
USB
Exp
1
2
3
Logidy


AKAI
professional
LPD8
LAPTOP PAD CONTROLLER
PROGRAM
PAD
PROG CHNG
CC
PAD 5
PAD 6
PAD 7
PAD 8
PAD 1
PROG 1
PAD 2
PROG 2
PAD 3
PROG 3
PAD 4
PROG 4
K1
K2
K3
K4
K5
K6
K7
K8
MIN
MAX


Puck.js
puck-js.com
open source
Get started at puck-js.com/start

THE SLIDE DECK

LOGIDY UMI3

Left Button -> 60 (NoteOn) -> Reveal.prev()
Right Button -> 64 (NoteOn) -> Reveal.next()
Middle Button -> 62 (NoteOn) -> triggerMedia()

Expression Pedal -> 0-127 (CC) -> (...)

THE SLIDE DECK

LOGIDY UMI3

```
midiInputs['Logidy UMI3'].addListener('noteon', '1', function(e){
  if (e.data[0] == 144 && e.data[2] == 64 ) {
    var value = e.data[1];
    if (value === 60) Reveal.prev();
    if (value === 64) Reveal.next()
    if (value === 62) triggerMedia();
  }
});
```

THE SLIDE DECK

LOGIDY UMI3

```
midiInputs['Logidy UMI3'].previous_value = 0;
midiInputs['Logidy UMI3'].addListener('controlchange', 'all', function(e){
  if (e.data[2] == 127 && e.data[2] > midiInputs['Logidy UMI3'].previous_value) {
    location.reload();
  }
  midiInputs['Logidy UMI3'].previous_value = e.data[2];
});
```

THE SLIDE DECK

PUCK.JS

One Click -> 80 (CC) -> Reveal.prev()
Double Click -> 81 (CC) -> Reveal.next()
Triple Click -> 82 (CC) -> triggerMedia()

THE SLIDE DECK

PUCK.JS

```
midiInputs['PuckCC Bluetooth'].addListener('controlchange', 11, function(e){
  if (e.data[2] == 127 ) {
    var value = e.data[1];
    if (value === 80) Reveal.next();
    if (value === 81) Reveal.prev()
    if (value === 82) triggerMedia();
  }
});
```

THE SLIDE DECK

PUCK.JS — ESPRUINO CODE

</>

```
const CHANNEL = 10;
const CONTROLLER = 80;
var clickcount = 0;
var clickevent = null;

setWatch(function(e) {
  clickcount++;

  if (clickevent !== null) clearTimeout(clickevent);

  if (clickcount === 1) {
    LED1.reset();
    LED2.set();
    LED3.reset();
  } else if (clickcount === 2) {
```

THE SLIDE DECK

LOGIDY UMI3 & PUCK.JS

triggerMedia()

</>

```
function triggerMedia() {
  var media = document.querySelector('section.present video, section.present audio');
  if (media.paused) {
    media.play();
  }else{
    media.pause();
  }
}
```

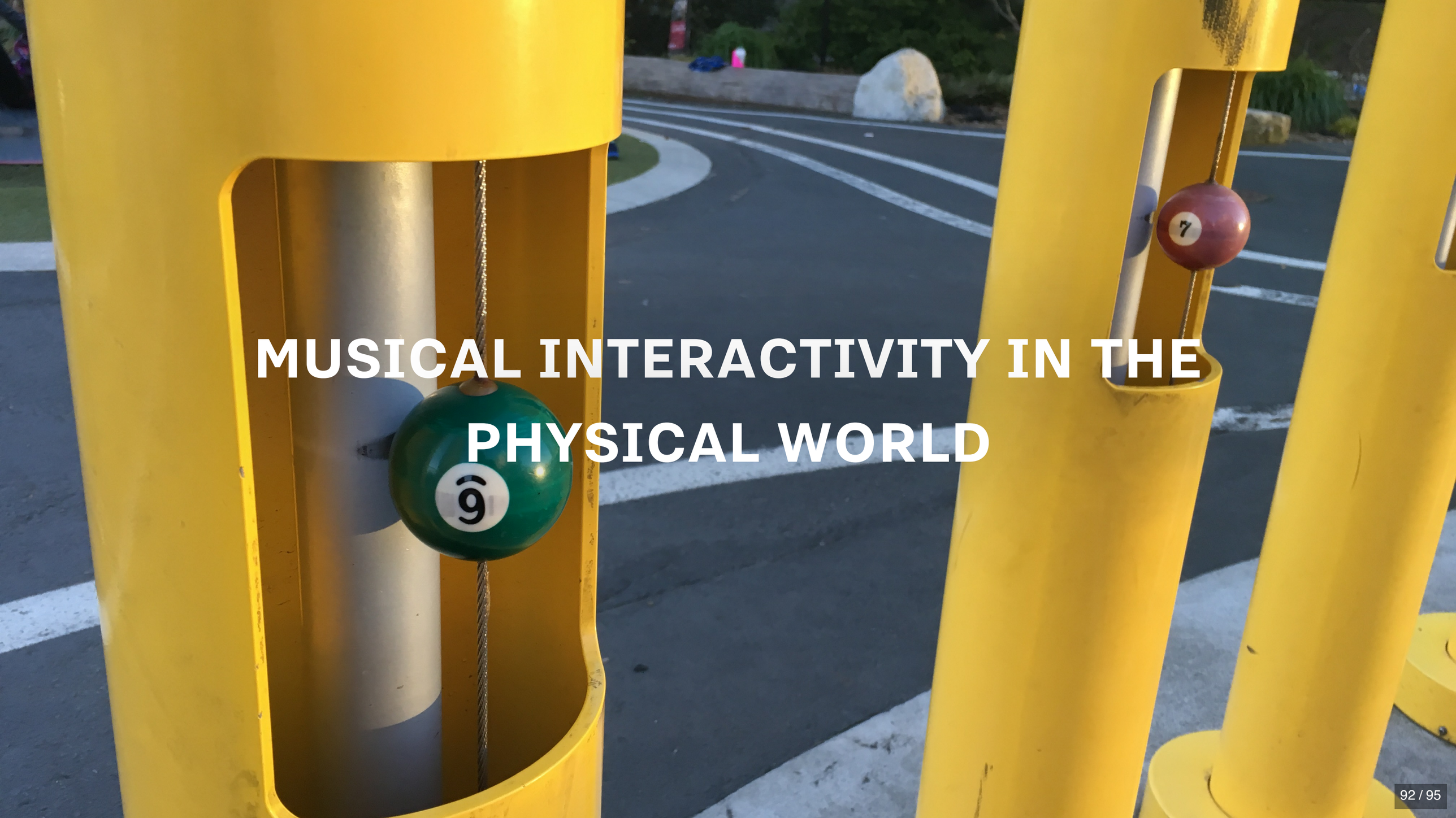
THE SLIDE DECK

LPD8

Knobs 1-8 -> 0-127 (CC) -> (...)

Only triggered when PROG4 is selected on device.


9
7


MUSICAL INTERACTIVITY IN THE
PHYSICAL WORLD


MUSICAL INTERACTIVITY IN THE
PHYSICAL WORLD

REPURPOSING THINGS FOR MUSIC

WHY NOT REPURPOSE MUSIC FOR OTHER THINGS?

СПАСИБО!

(THANK YOU!)

George Mandis

Email: george@mand.is Twitter: @georgeMandis

GitHub: georgemandis

Website: george.mand.is

More on MIDI and credits: midi.mand.is