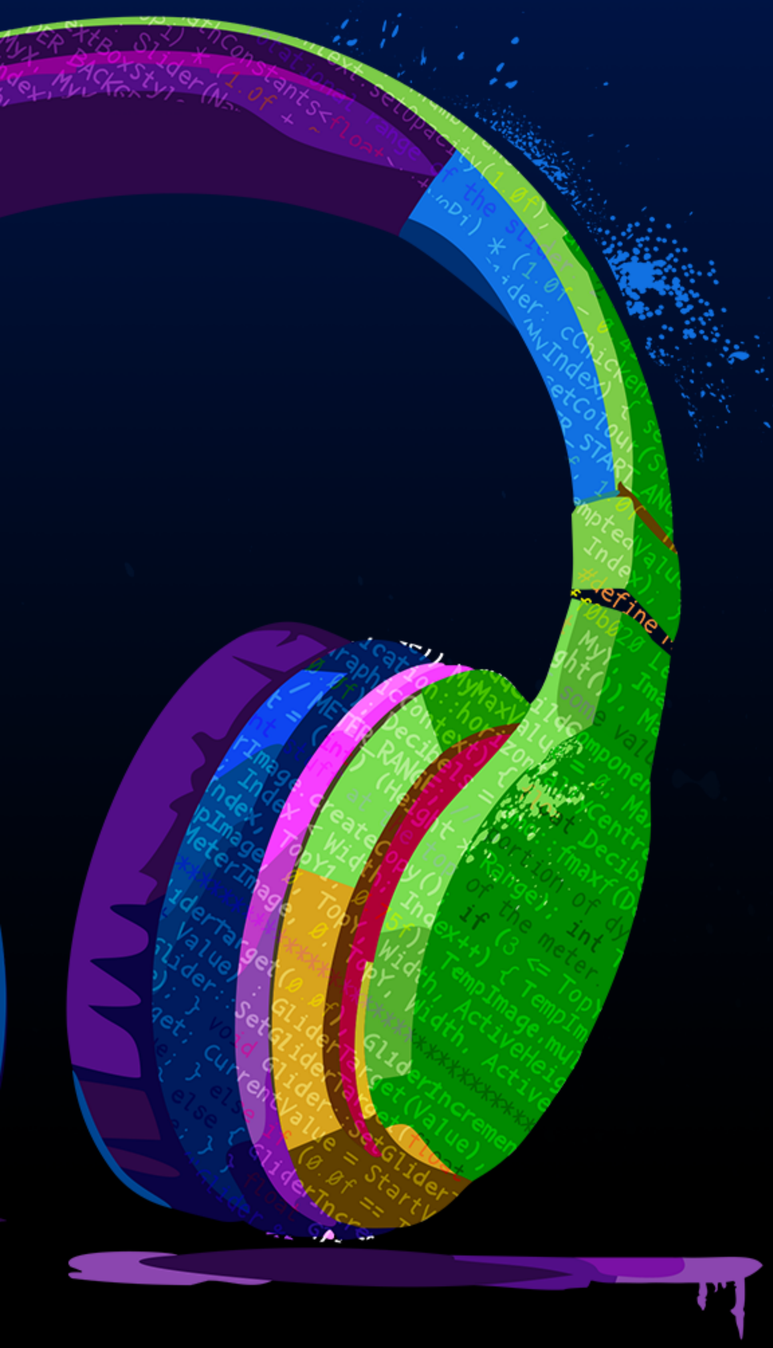


ADC24 Bristol

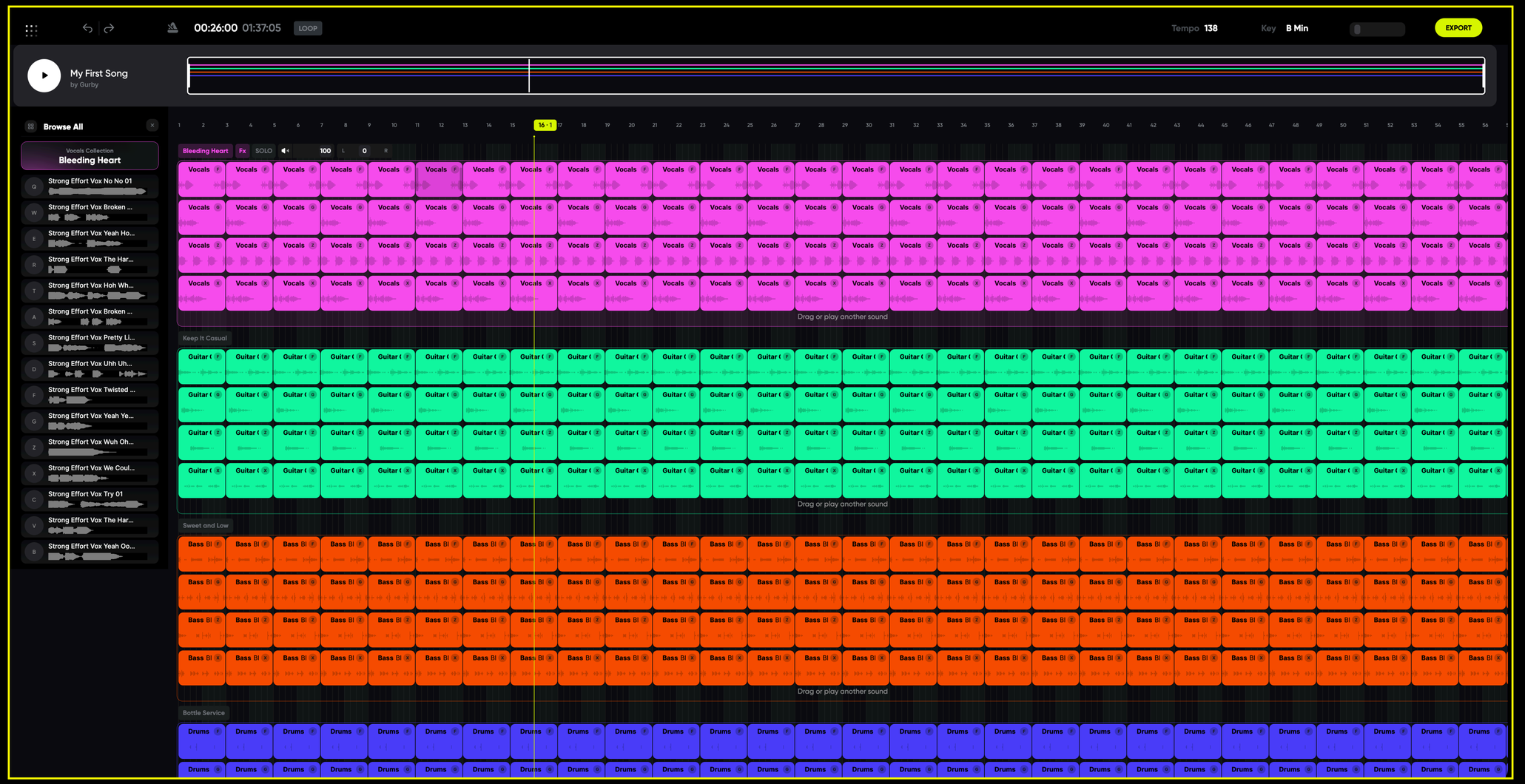
HIGH PERFORMANCE WEBVIEW USER INTERFACES

GETTING THE MOST OUT OF THE 2D CANVAS AND DOM

ARTHUR CARABOTT

OUTPUT

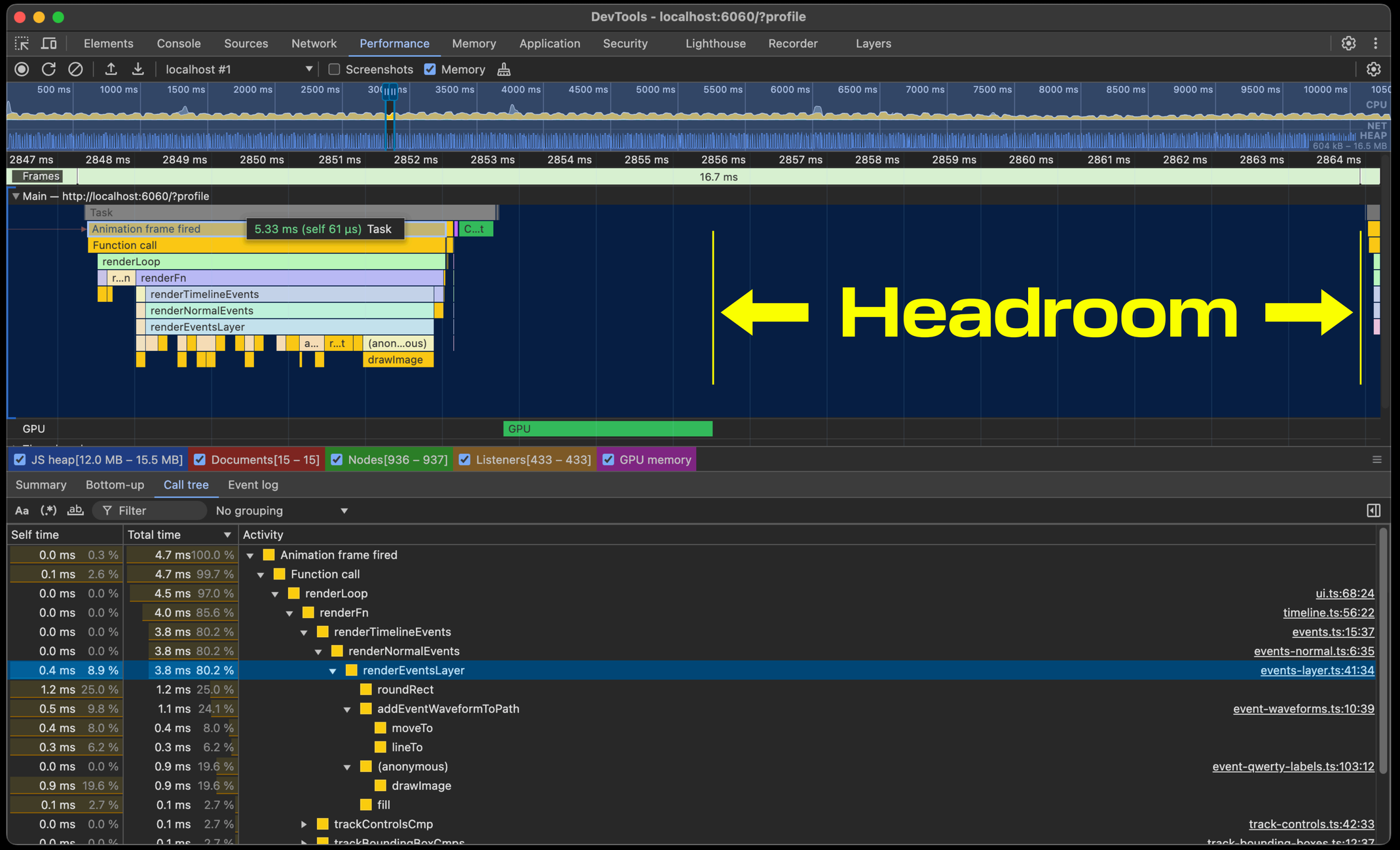
OUTPUT CREATOR

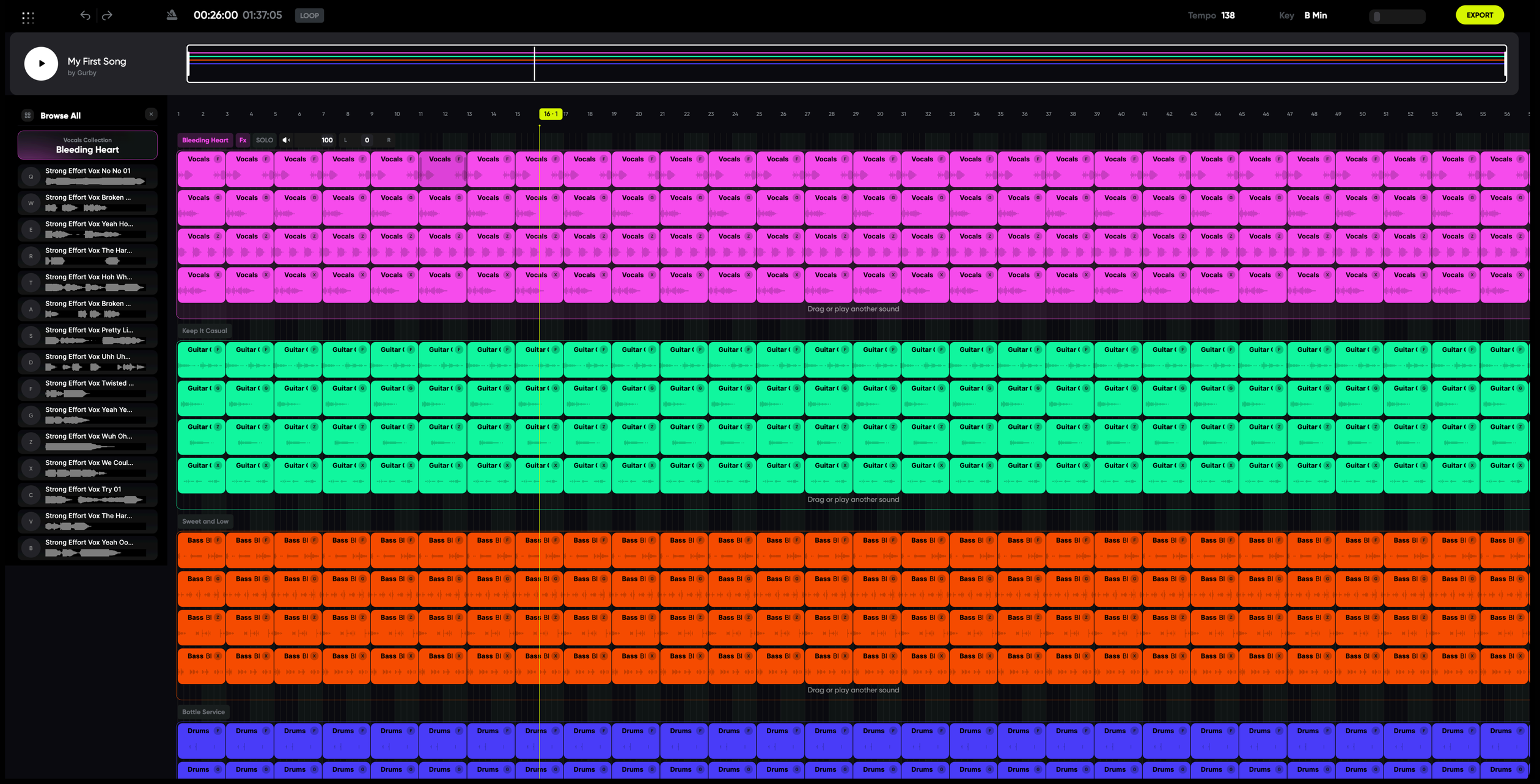


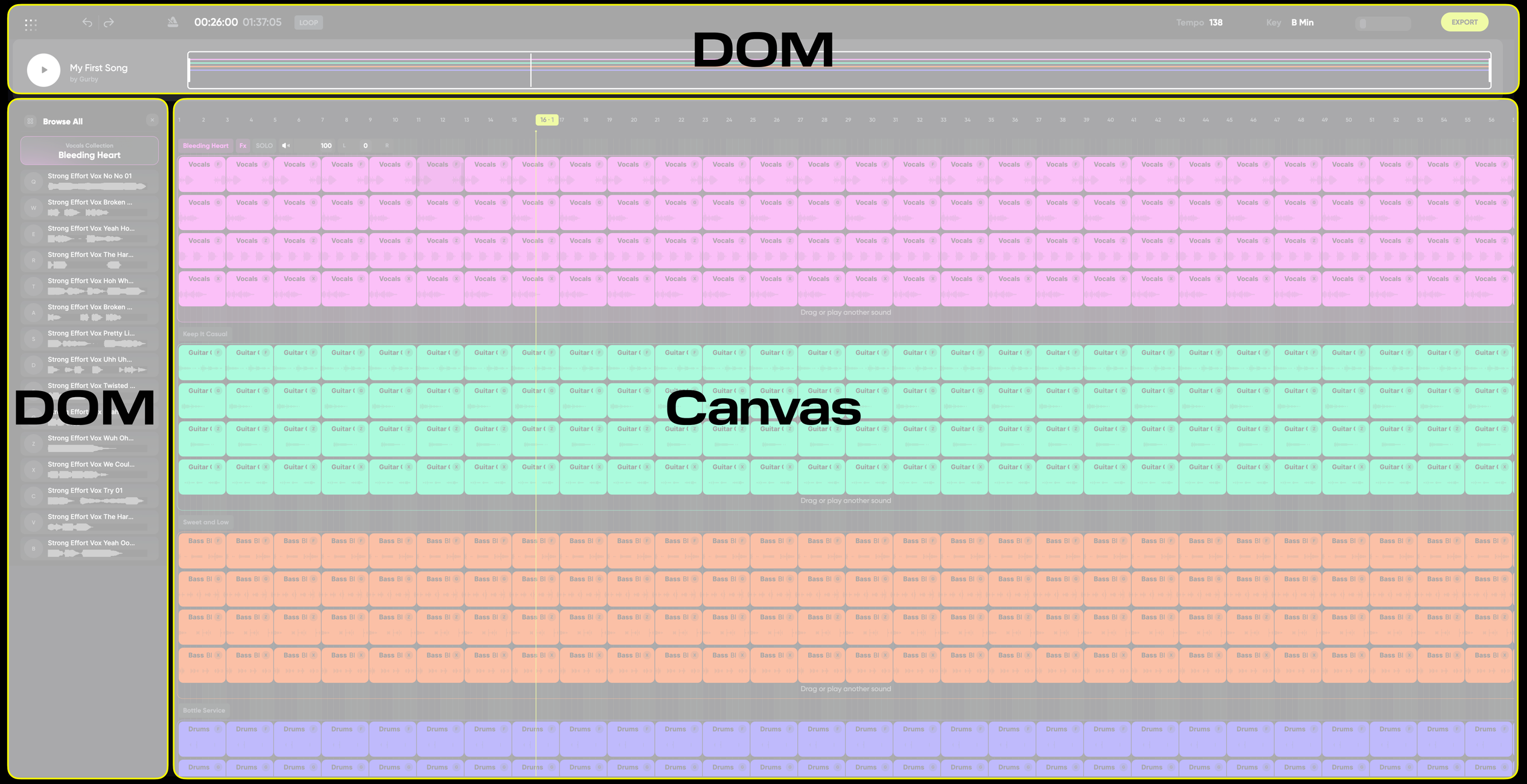

Render Duration:
0.00 ms (0.00 - 0.00 ms)
0 fps (0 - 0 fps)
Actions:
Scrolling, Dragging, Zooming
Visible Regions:
1710 (max: 1710)
Screen:
2560 x 1440 @ 2x
My First Song
by Gurby
Browse All
Vocals Collection
Bleeding Heart
Strong Effort Vox No No 01
Strong Effort Vox Broken ...
Strong Effort Vox Yeah Ho...
Strong Effort Vox The Har...
Strong Effort Vox Hoh Wh...
Strong Effort Vox Broken ...
Strong Effort Vox Pretty Li...
Strong Effort Vox Uhh Uh...
Strong Effort Vox Twisted ...
Strong Effort Vox Yeah Ye...
Strong Effort Vox Wuh Oh...
Strong Effort Vox We Coul...
Strong Effort Vox Try 01
Strong Effort Vox The Har...
Strong Effort Vox Yeah Oo...
Bleeding Heart
Fx
SOLO
Drag or play another sound
Keep It Casual
Drag or play another sound
Sweet and Low
Drag or play another sound
Bottle Service
MacBook Pro
13-inch, M1, 2020
Chip Apple M1
Memory 8 GB
Startup disk Macintosh HD
Serial number C02FG1P0Q05H
macOS Sonoma 14.5
More Info...
Regulatory Certification
™ and © 1983-2024 Apple Inc.
All Rights Reserved

Render Duration:

```
5.00 ms  (1.90 - 8.60 ms)
 200 fps ( 116 -  526 fps)
```


DevTools - localhost:6060/?profile
Elements
Console
Sources
Network
Performance
Memory
Application
Security
Lighthouse
Recorder
Layers
localhost #1
Screenshots
Memory
CPU
NET
HEAP
604 kB – 16.5 MB
Frames
16.7 ms
Main — http://localhost:6060/?profile
Task
Animation frame fired
5.33 ms (self 61 µs) Task
Function call
renderLoop
renderFn
renderTimelineEvents
renderNormalEvents
renderEventsLayer
drawImage
Headroom
GPU
JS heap[12.0 MB – 15.5 MB]
Documents[15 – 15]
Nodes[936 – 937]
Listeners[433 – 433]
GPU memory
Summary
Bottom-up
Call tree
Event log
Filter
No grouping
Self time
Total time
Activity
Animation frame fired
Function call
renderLoop
renderFn
renderTimelineEvents
renderNormalEvents
renderEventsLayer
roundRect
addEventWaveformToPath
moveTo
lineTo
(anonymous)
drawImage
fill
trackControlsCmp
ui.ts:68:24
timeline.ts:56:22
events.ts:15:37
events-normal.ts:6:35
events-layer.ts:41:34
event-waveforms.ts:10:39
event-qwerty-labels.ts:103:12
track-controls.ts:42:33


00:26:00 01:37:05
LOOP
Tempo 138
Key B Min
EXPORT
My First Song
by Gurby
Browse All
Vocals Collection
Bleeding Heart
Strong Effort Vox No No 01
Strong Effort Vox Broken ...
Strong Effort Vox Yeah Ho...
Strong Effort Vox The Har...
Strong Effort Vox Hoh Wh...
Strong Effort Vox Broken ...
Strong Effort Vox Pretty Li...
Strong Effort Vox Uhh Uh...
Strong Effort Vox Twisted ...
Strong Effort Vox Yeah Ye...
Strong Effort Vox Wuh Oh...
Strong Effort Vox We Coul...
Strong Effort Vox Try 01
Strong Effort Vox The Har...
Strong Effort Vox Yeah Oo...
Bleeding Heart
Fx
SOLO
Drag or play another sound
Keep It Casual
Drag or play another sound
Sweet and Low
Drag or play another sound
Bottle Service


00:26:00 01:37:05
LOOP
Tempo 138
Key B Min
EXPORT
My First Song
by Gurby
DOM
Browse All
Vocals Collection
Bleeding Heart
Strong Effort Vox No No 01
Strong Effort Vox Broken ...
Strong Effort Vox Yeah Ho...
Strong Effort Vox The Har...
Strong Effort Vox Hoh Wh...
Strong Effort Vox Broken ...
Strong Effort Vox Pretty Li...
Strong Effort Vox Uhh Uh...
Strong Effort Vox Twisted ...
Strong Effort Vox Wuh Oh...
Strong Effort Vox We Coul...
Strong Effort Vox Try 01
Strong Effort Vox The Har...
Strong Effort Vox Yeah Oo...
DOM
Bleeding Heart
Fx
SOLO
Drag or play another sound
Keep It Casual
Drag or play another sound
Sweet and Low
Drag or play another sound
Bottle Service
Canvas

OUTPUT

DOM

```
<section>
    <button>Toggle Metronome</button>

    <label>Tempo: </label>

    <input type="range" min="1" max="999" />
</section>
```

HTML

```html
<section>

    <button>
        Toggle Metronome
    </button>

    <label>
        Tempo:
    </label>

    <input
        type="range"
        min="1"
        max="999"
    />

</section>
```

JavaScript

```js
const topEl = document.createElement("section");

{
    const metronomeEl = document.createElement("button");
    metronomeEl.textContent = "Toggle Metronome";
    topEl.appendChild(metronomeEl);
}

{
    const tempoLabelEl = document.createElement("label");
    tempoLabelEl.textContent = "Tempo:";

    const tempoEl = document.createElement("input");
    tempoEl.type = "range";
    tempoEl.min = "1";
    tempoEl.max = "999";
    topEl.appendChild(tempoLabelEl);
}
```

Canvas

```
ctx.beginPath();
ctx.roundRect( 10,  10, 400, 200, 10);
ctx.roundRect(400, 220, 200, 100, 10);

ctx.fillStyle = "rgb(43, 156, 212)";
ctx.fill();
```

"When performance matters, Canvas beats DOM"

Me
This talk, today

1. **Canvas**
2. DOM
3. Layering

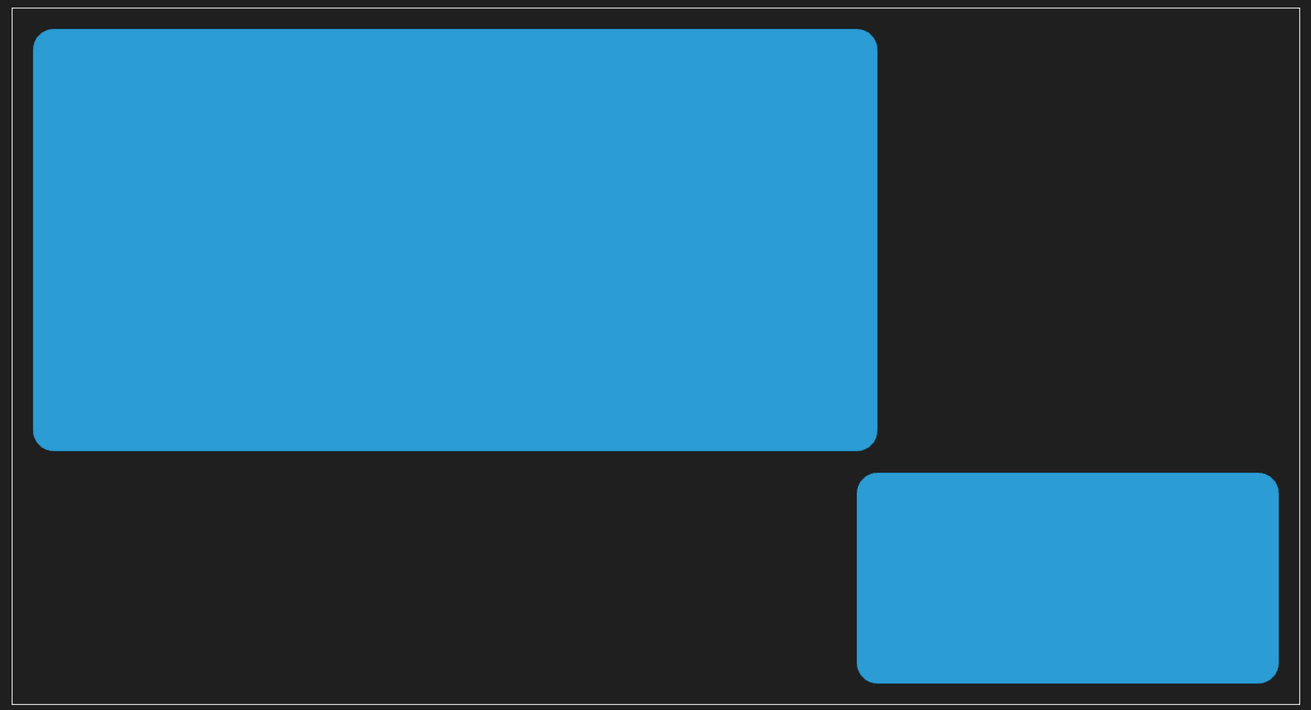

<canvas>

2D

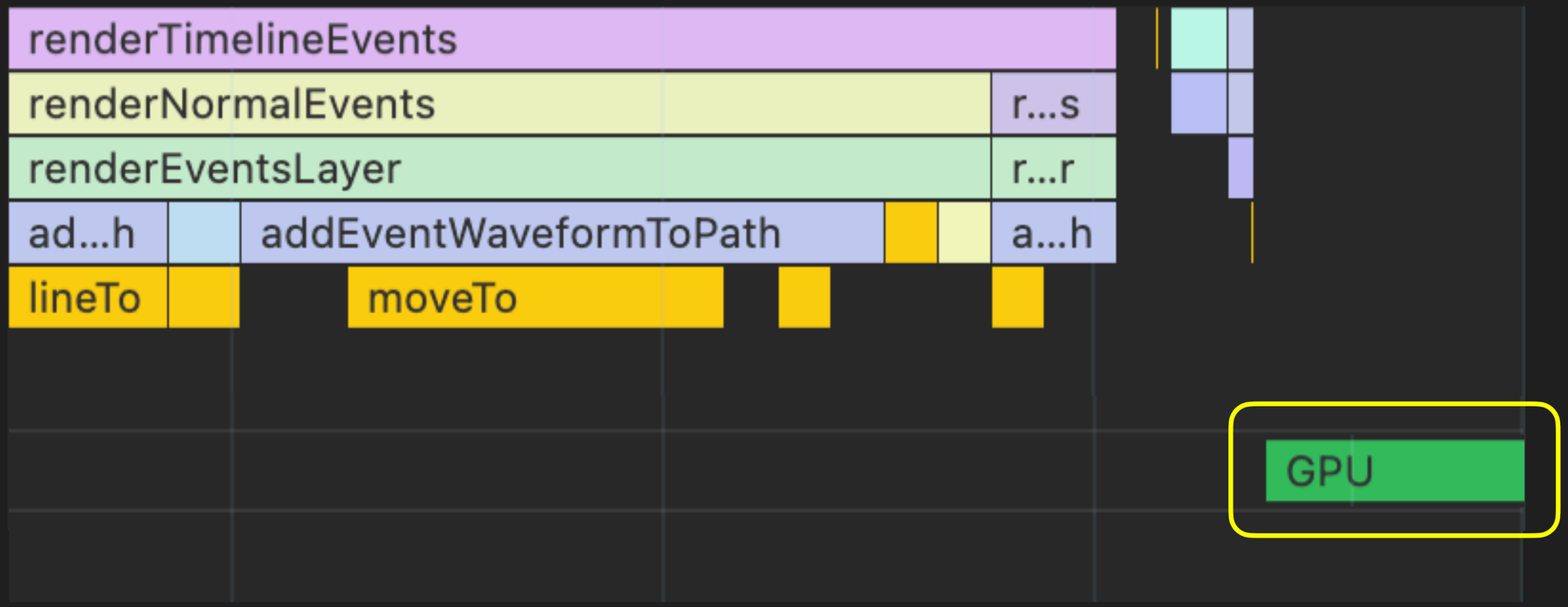
WebGPU

2D

```
ctx.beginPath();
ctx.roundRect( 10,  10, 400, 200, 10);
ctx.roundRect(400, 220, 200, 100, 10);

ctx.fillStyle = "rgb(43, 156, 212)";
ctx.fill();
```



2D



Hardware Accelerated

Tradeoffs:

Performance 

Control 

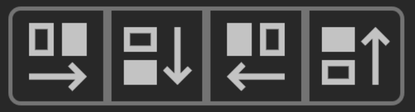
Layout 

Animations 

Interactions 

Accessibility 

flex-direction: row



flex-wrap: nowrap

align-content: normal

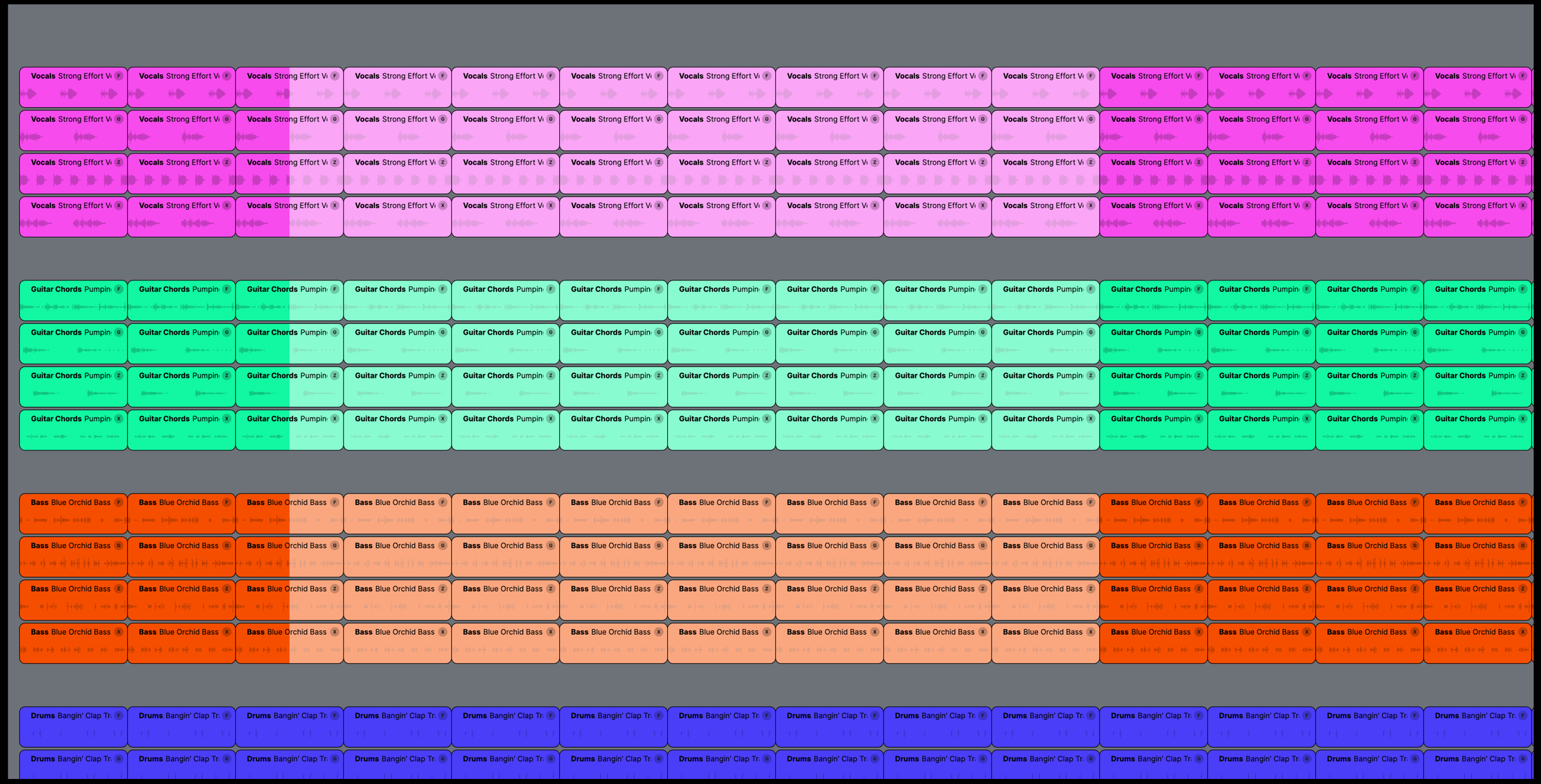
justify-content: normal

align-items: normal

"It is very unlikely in any given situation that one thing exists in a vacuum.

The most common case is I have multiple, so I may as well transform them together"

Mike Acton
CppCon 2014: "Data-Oriented Design and C++" 37m 30s
https://www.youtube.com/watch?v=rX0ItVEVjHc


Vocals Strong Effort Vo F
Vocals Strong Effort Vo G
Vocals Strong Effort Vo Z
Vocals Strong Effort Vo X
Guitar Chords Pumpin F
Guitar Chords Pumpin G
Guitar Chords Pumpin Z
Guitar Chords Pumpin X
Bass Blue Orchid Bass F
Bass Blue Orchid Bass G
Bass Blue Orchid Bass Z
Bass Blue Orchid Bass X
Drums Bangin' Clap Tr F
Drums Bangin' Clap Tr G

Slow: individual rendering

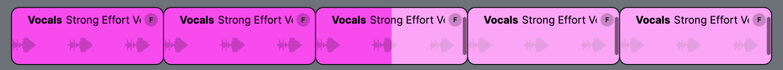
```
const renderEvent = (ctx: CanvasRenderingContext2D, event: TimelineEvent) => {
    {…
    }
};

export const renderEvents = (ctx: CanvasRenderingContext2D, events: TimelineEvent[]) => {
    for (const event of events) {
        renderEvent(ctx, event);
    }
};
```

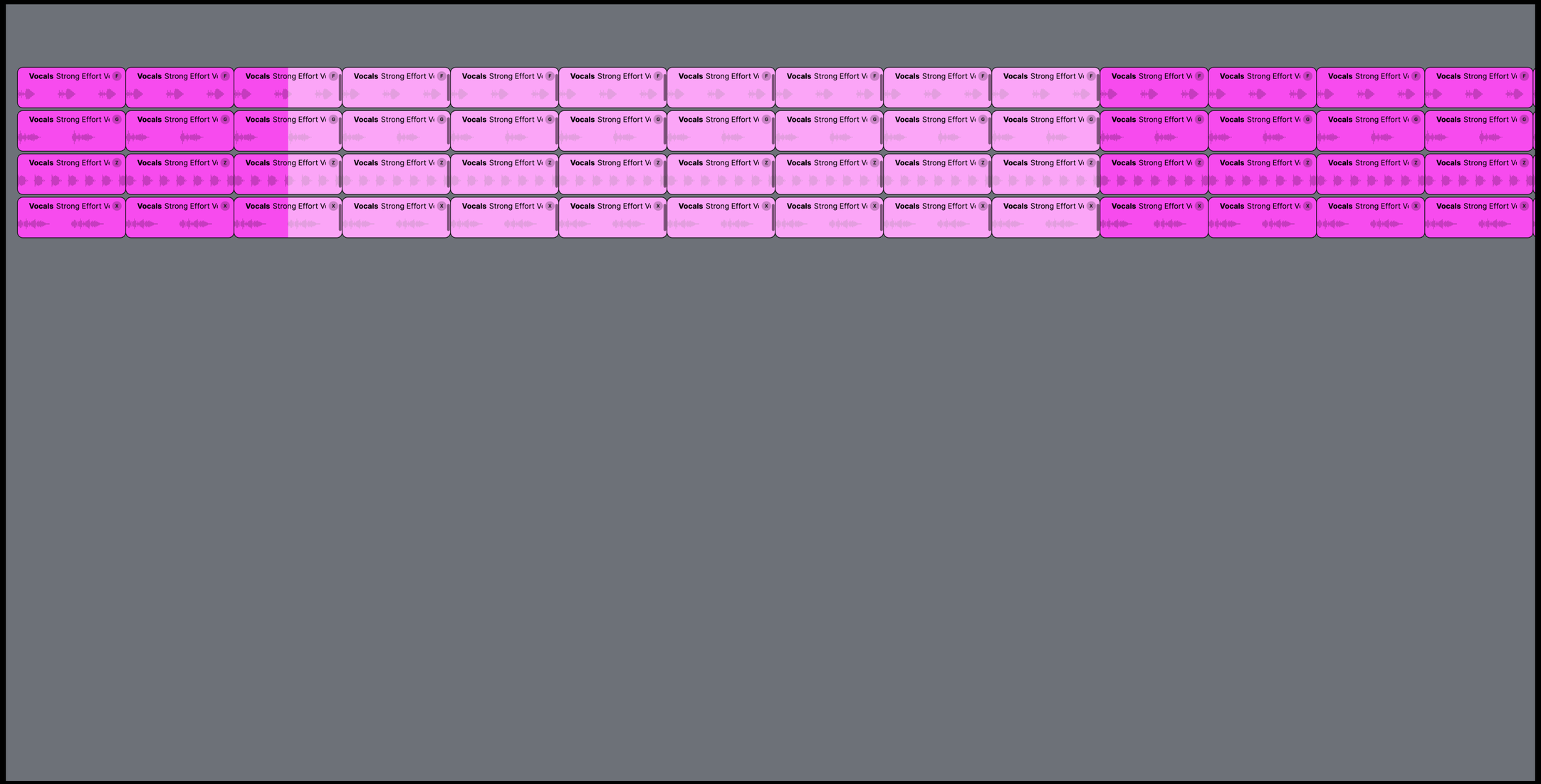

Vocals Strong Effort V

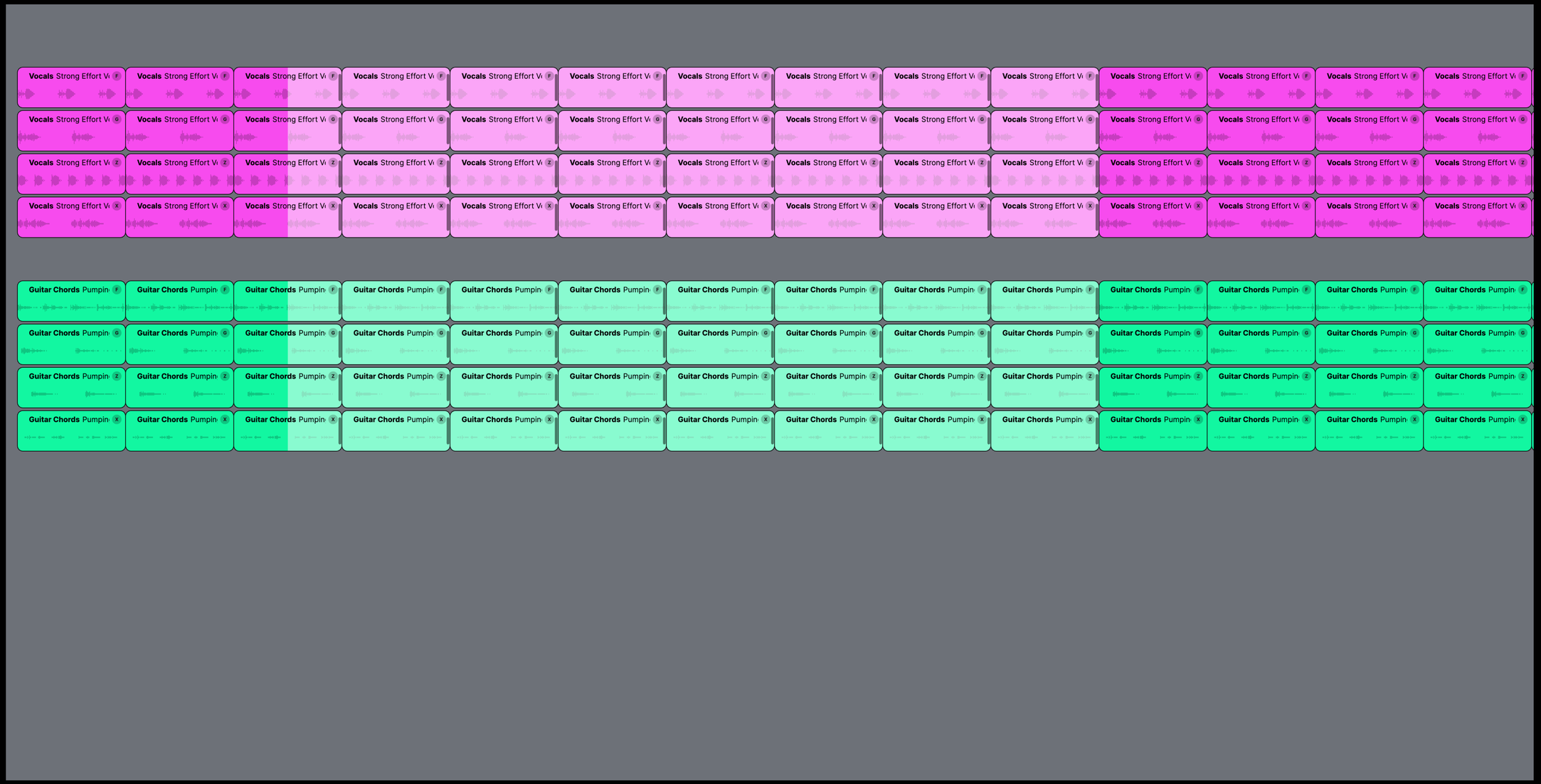

Vocals Strong Effort V
Vocals Strong Effort V


Vocals Strong Effort V
Vocals Strong Effort V
Vocals Strong Effort V


Vocals Strong Effort V
Vocals Strong Effort V
Vocals Strong Effort V
Vocals Strong Effort V


Vocals Strong Effort V
Vocals Strong Effort V
Vocals Strong Effort V
Vocals Strong Effort V
Vocals Strong Effort V


Vocals Strong Effort V
F
G
Z
X

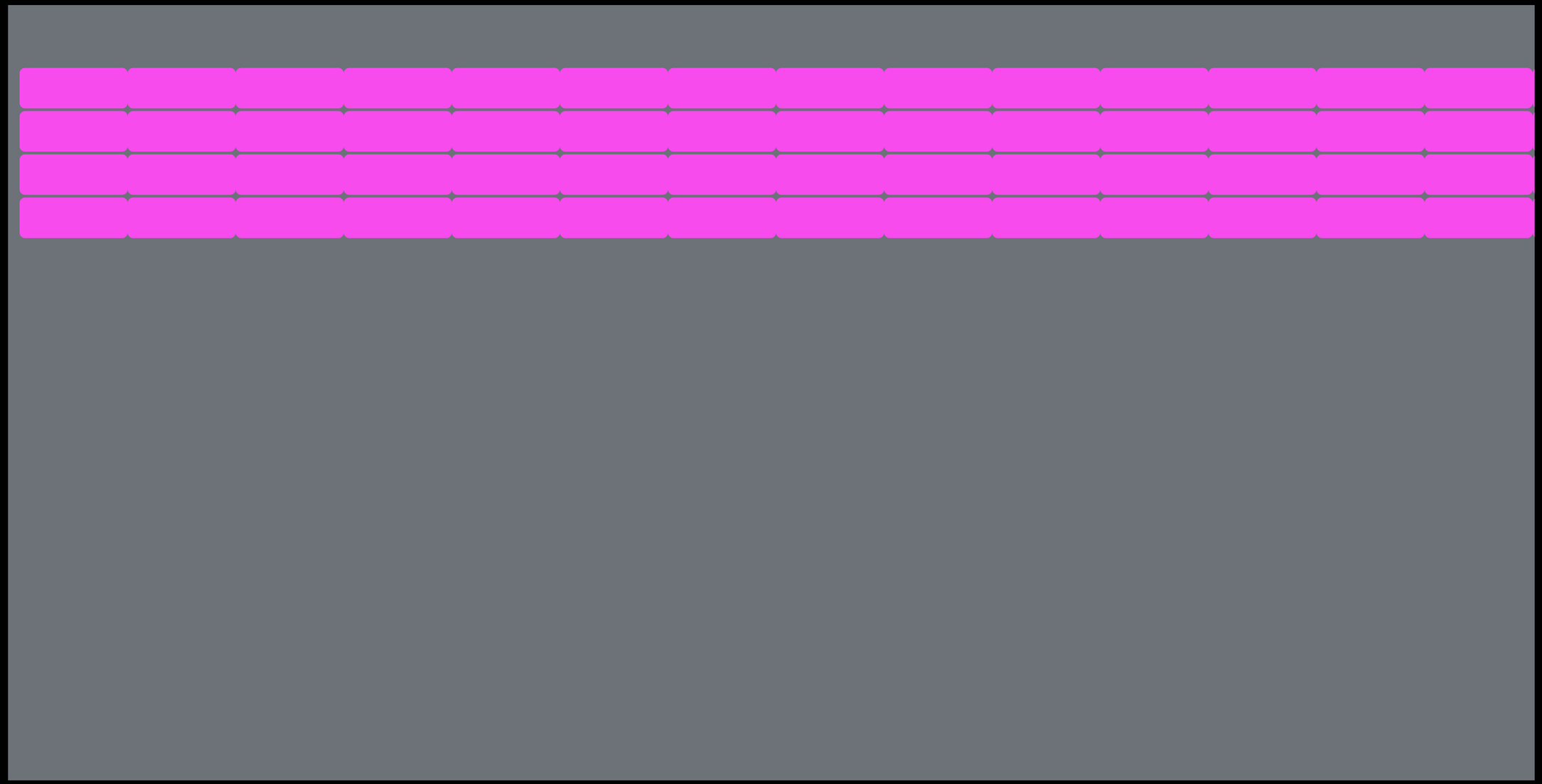


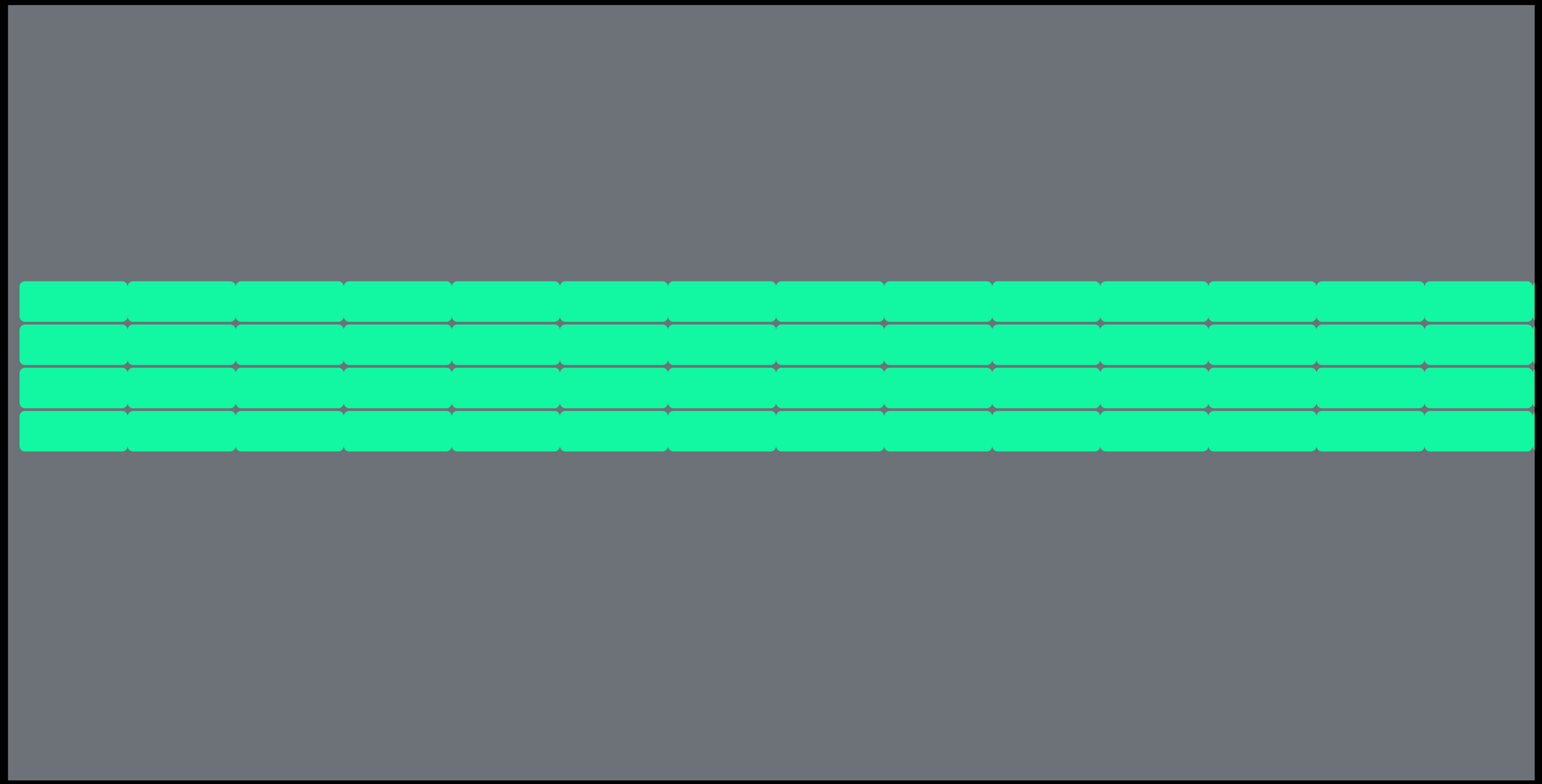
OUTPUT

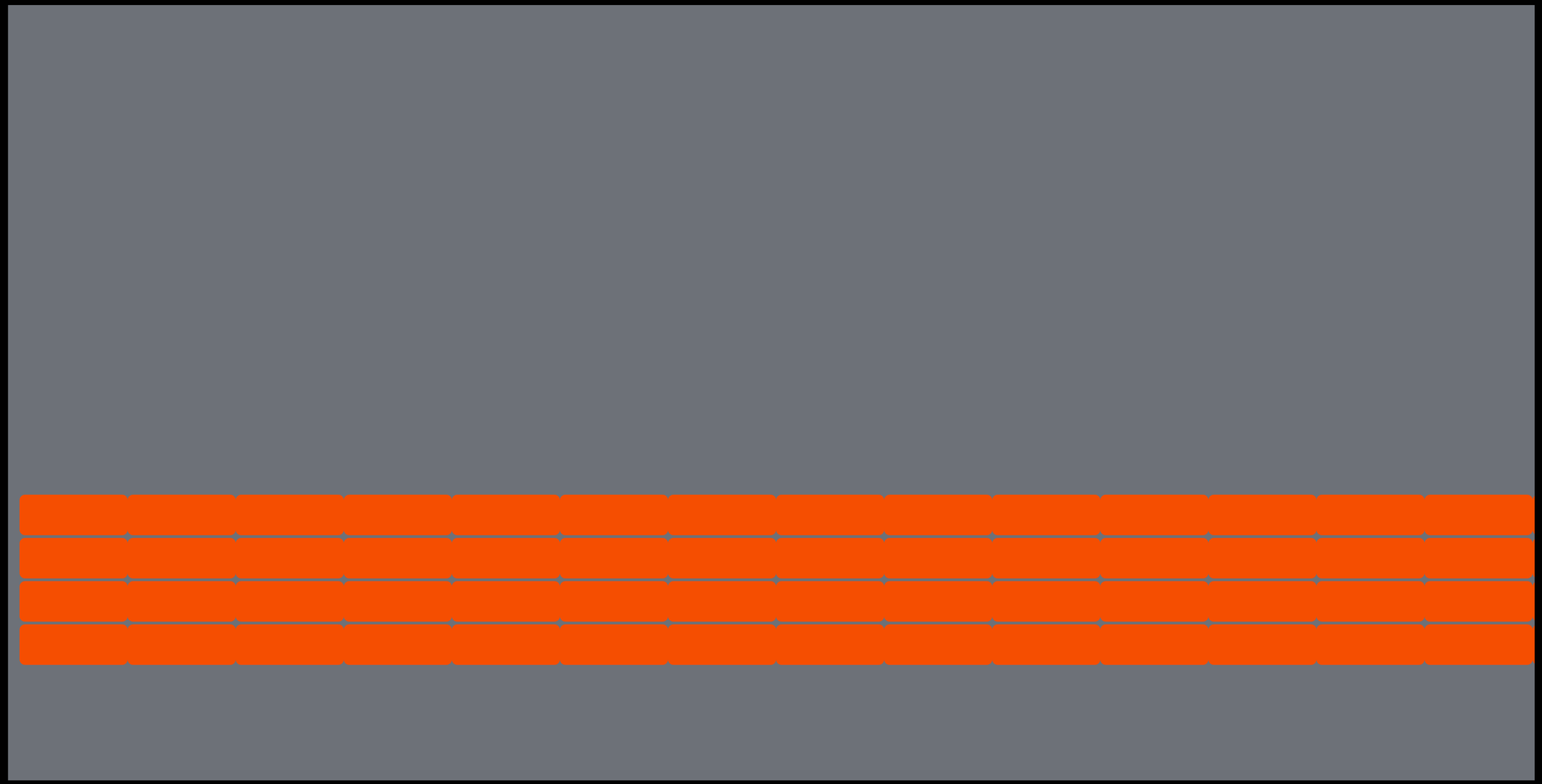

Vocals Strong Effort V
Guitar Chords Pumpin
Bass Blue Orchid Bass

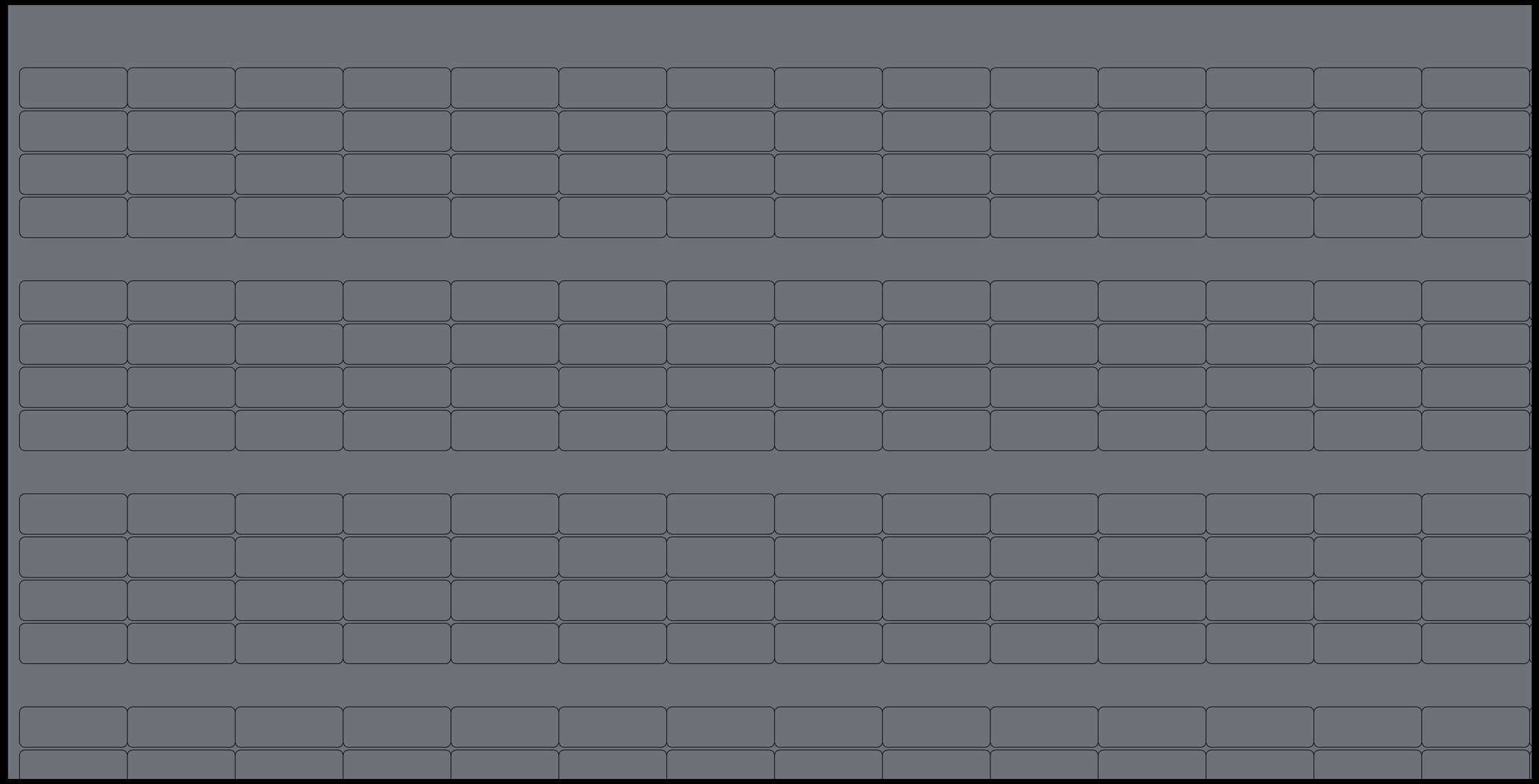

Vocals Strong Effort Vo
Guitar Chords Pumpin
Bass Blue Orchid Bass
Drums Bangin' Clap Tr

Fast: batch rendering

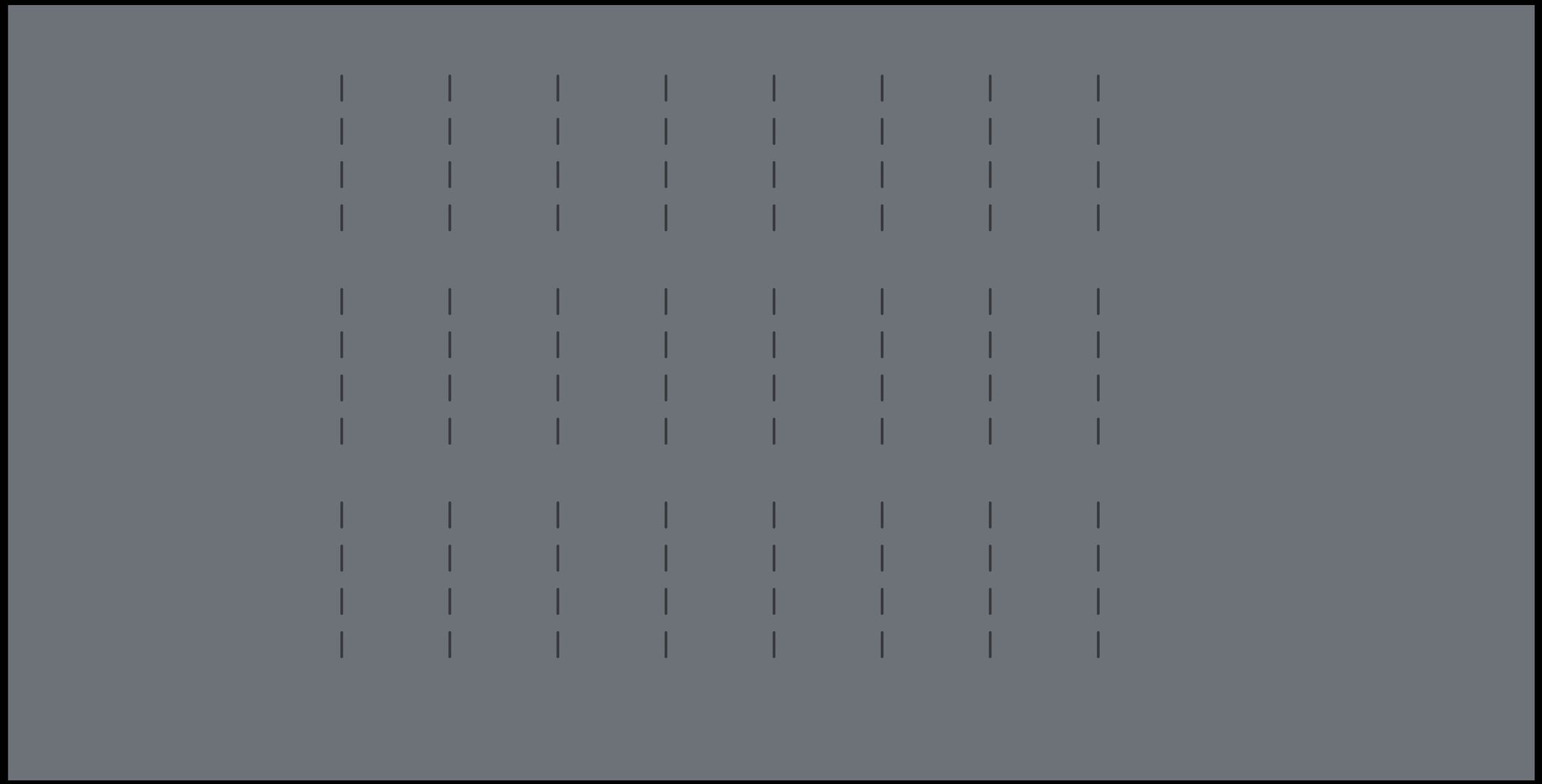


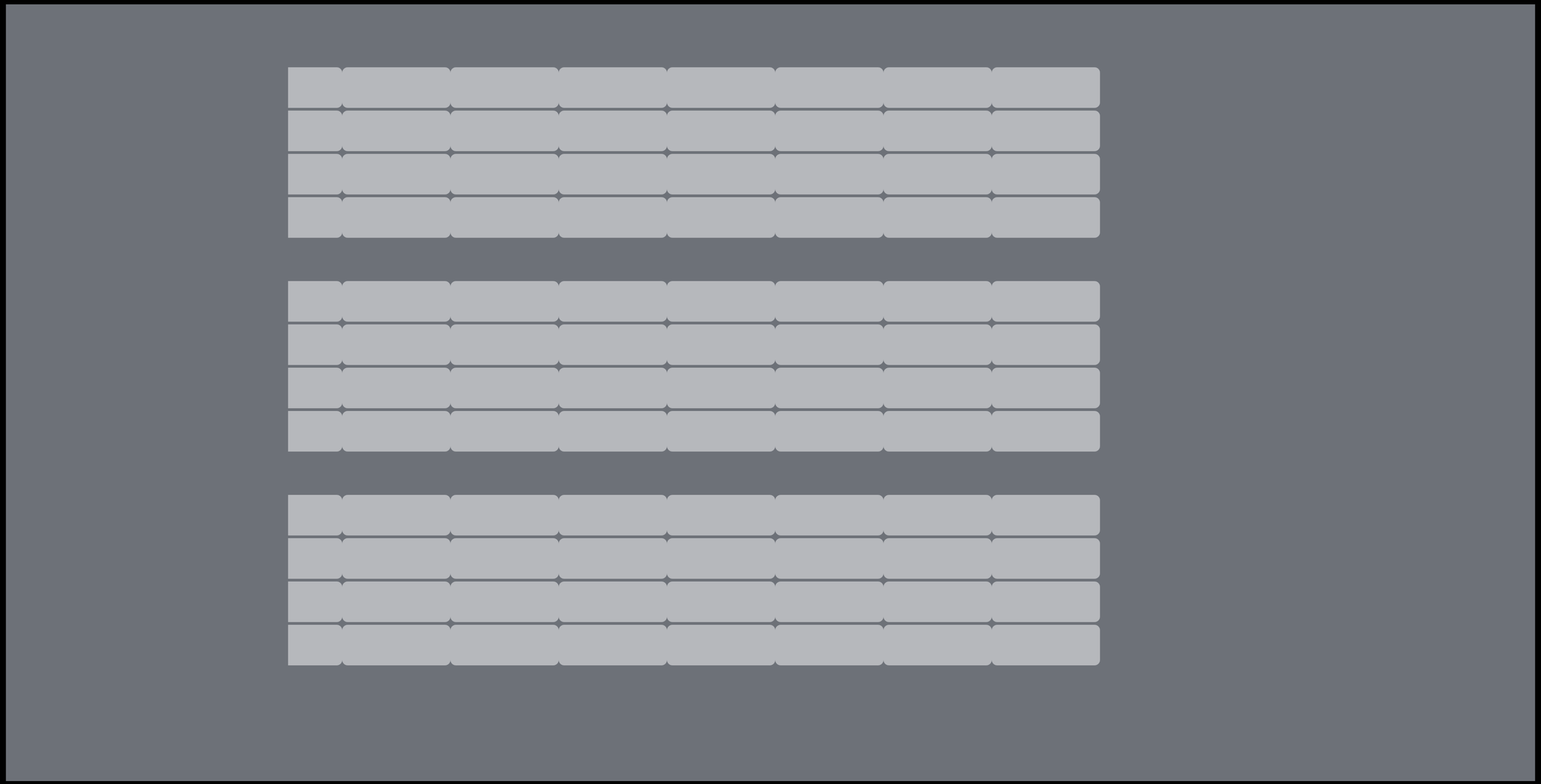


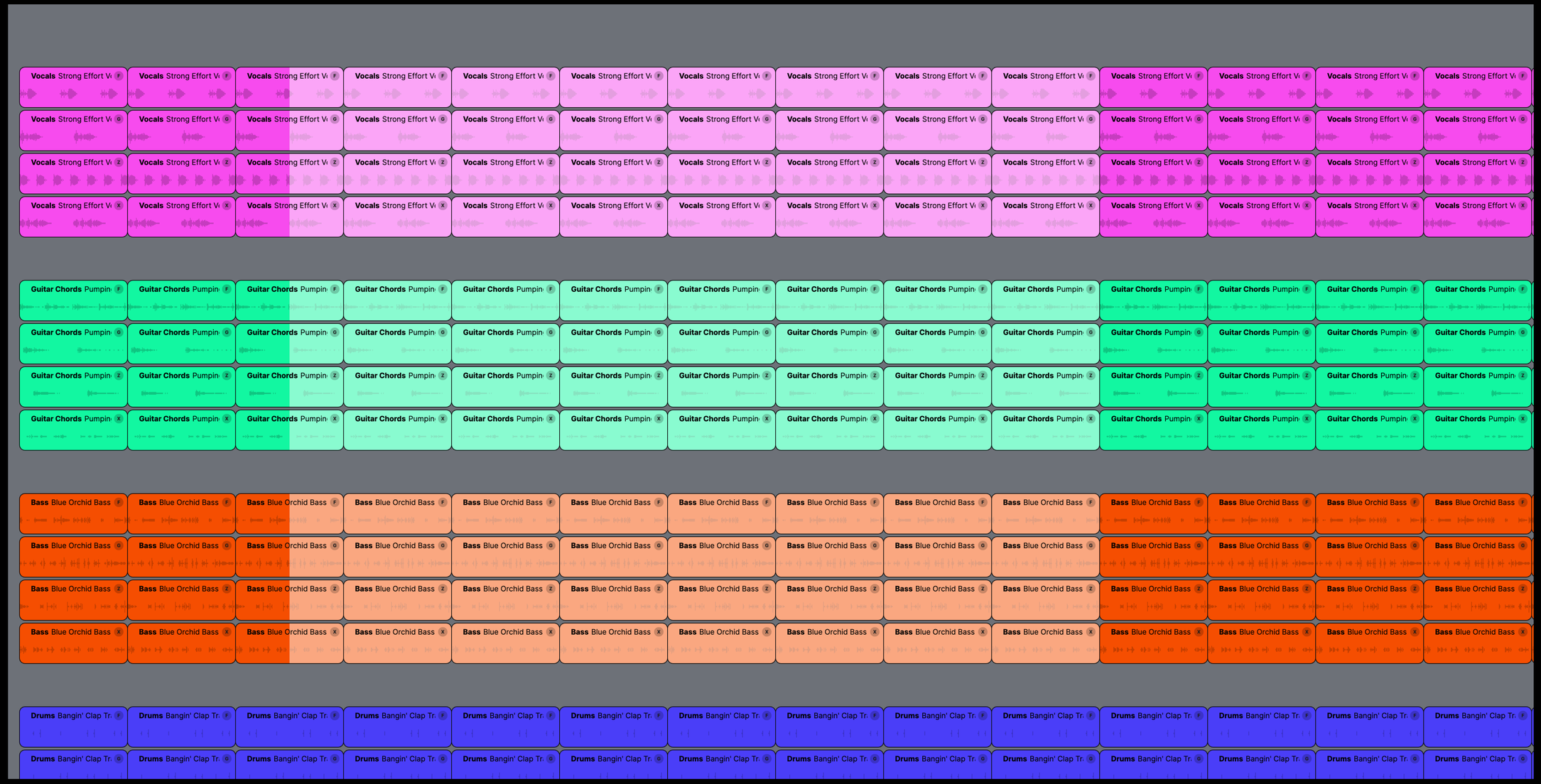





Vocals Strong Effort V
Guitar Chords Pumpin
Bass Blue Orchid Bass
Drums Bangin' Clap Tr



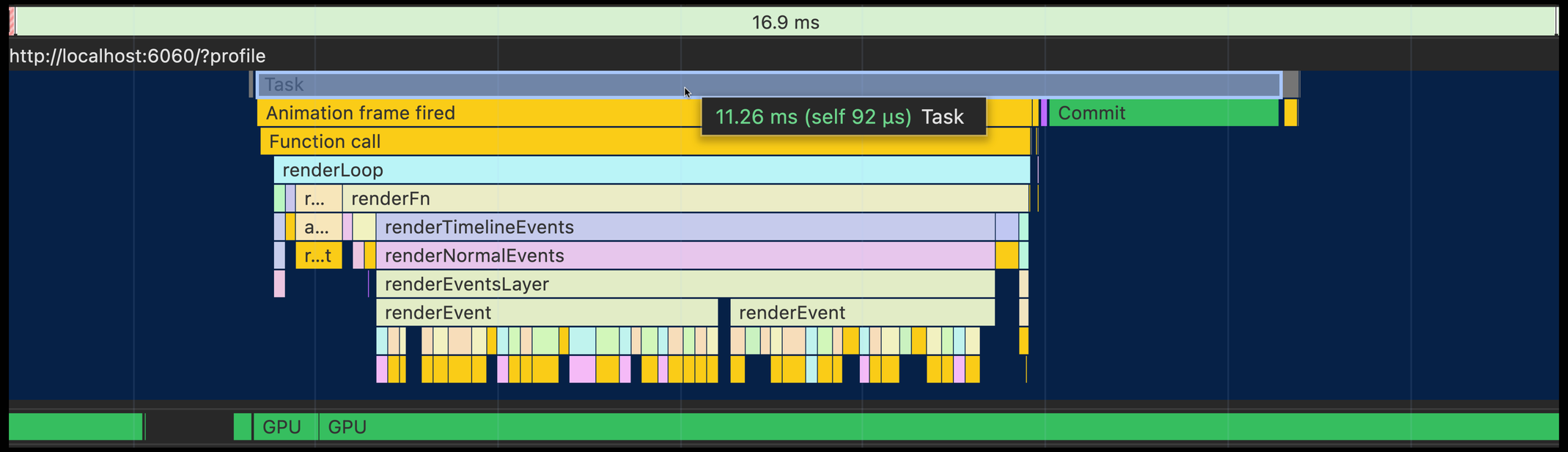



Vocals Strong Effort Vo
Guitar Chords Pumpin
Bass Blue Orchid Bass
Drums Bangin' Clap Tr

```ts
const renderEvents = (ctx: CanvasRenderingContext2D, events: TimelineEvent[]) => {
    // ---------------------- 1. create paths and view models ----------------------
    const allEventsPath = new Path2D();
    // [...]

    // ---------------------- 2. shared calculations -------------------------------
    const pixelsPerBeat = calculatePixelsPerBeat(zoomLevel);
    // [...]

    // ---------------------- 3. add to paths and view models ----------------------
    for (const event of events) {
        const bounds = createEventBounds(event, pixelsPerBeat);
        allEventsPath.roundRect(bounds.x, bounds.y, bounds.w, bounds.h, kEventRadii_px);
        // [...]
    }

    // ---------------------- 4. rendering -----------------------------------------
    ctx.fillStyle = "pink";
    ctx.fill(allEventsPath);
    // [...]
};
```

Slow

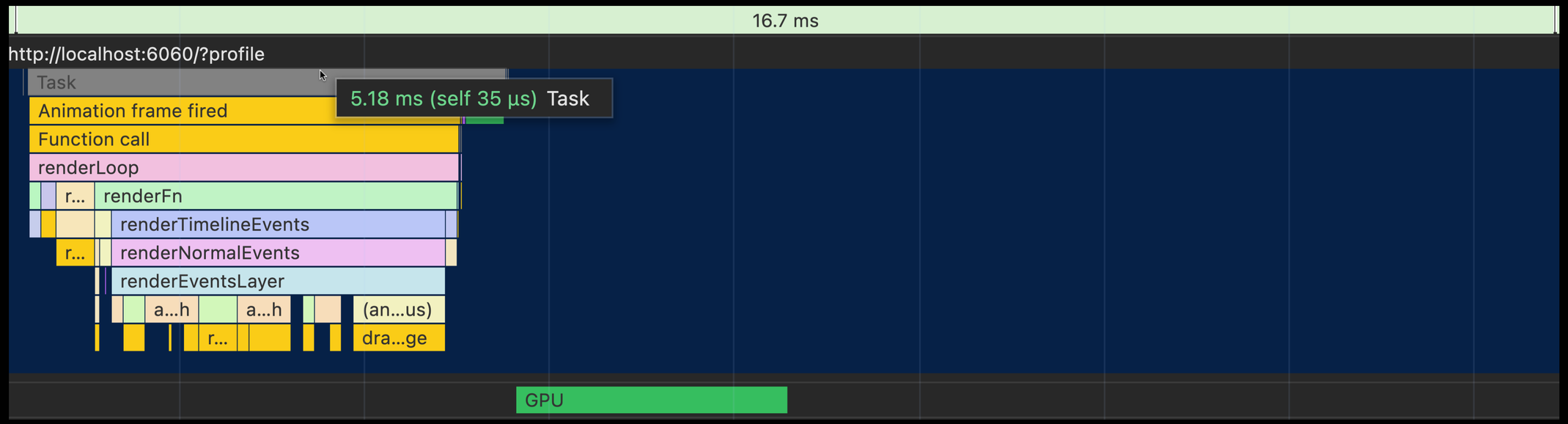
11.26ms



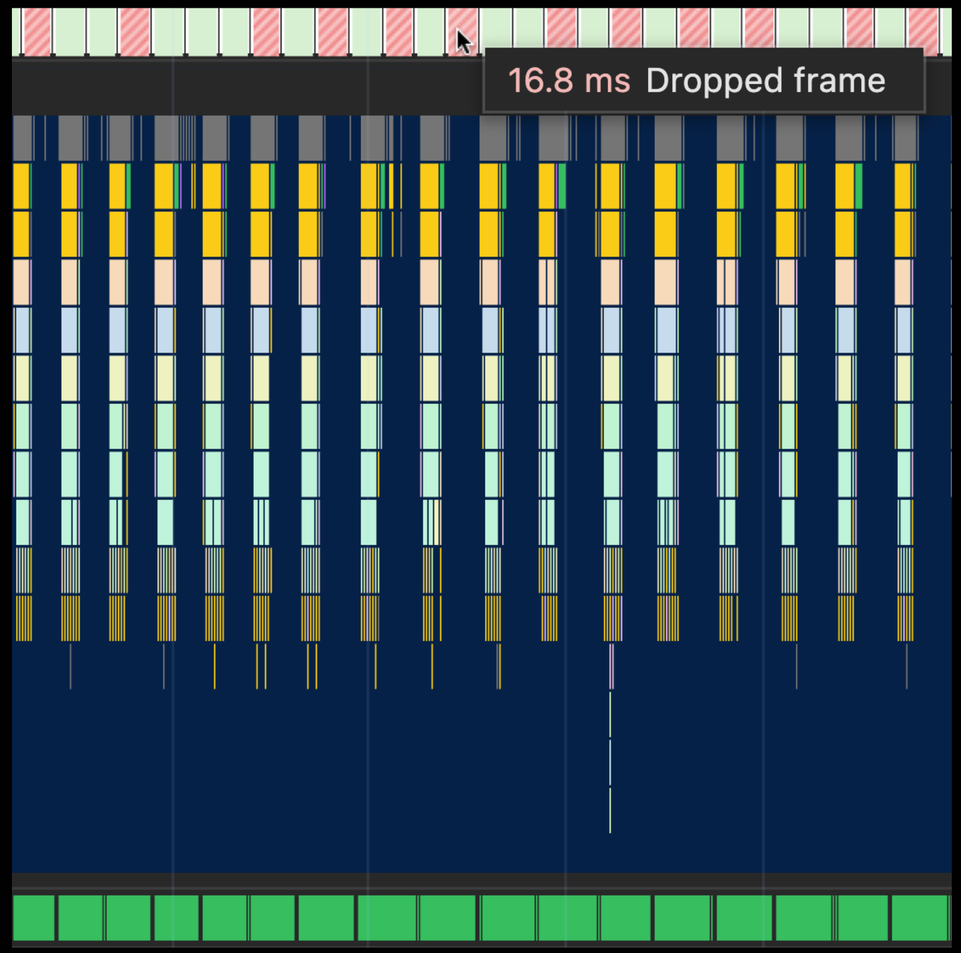
Fast

5.18ms

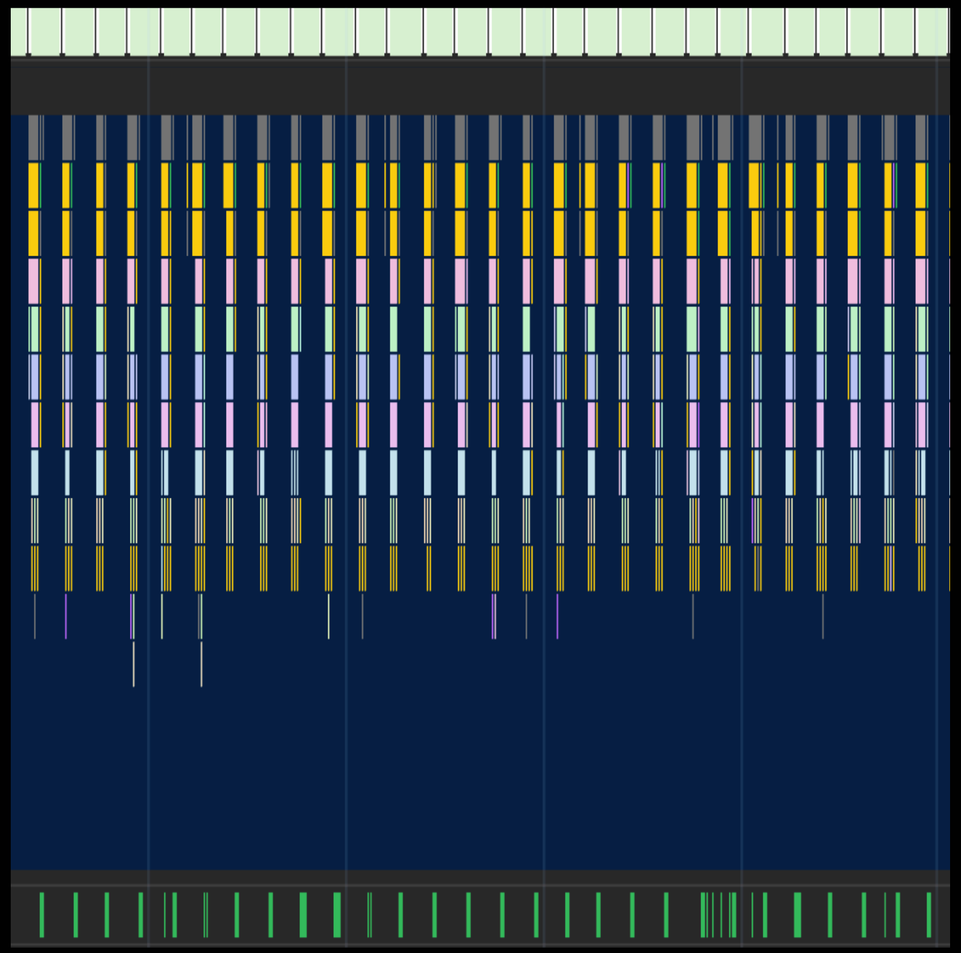
(2.2x)



Slow



Fast



Why?

Changing global properties

```
// event 1
ctx.fillStyle = "pink";

// event 2
ctx.fillStyle = "blue";

// event 3
ctx.fillStyle = "pink";
```

JavaScript

```
ctx.fillStyle = "pink";
```

C++ (64 LOC)

```
void BaseRenderingContext2D::setFillStyle(v8::Isolate* isolate,
                                          v8::Local<v8::Value> value,
                                          ExceptionState& exception_state) {
  ValidateStateStack();

  CanvasRenderingContext2DState& state = GetState();
  // This block is similar to that in setStrokeStyle(), see comments there for
  // details on this.
  if (value->IsString()) {
    v8::Local<v8::String> v8_string = value.As<v8::String>();
    UpdateIdentifiabilityStudyBeforeSettingStrokeOrFill(
        v8_string, CanvasOps::kSetFillStyle);
    if (state.IsUnparsedFillColor(v8_string)) {
      return;
    }
    Color parsed_color = Color::kTransparent;
    if (!ExtractColorFromV8StringAndUpdateCache(
            isolate, v8_string, exception_state, parsed_color)) {
      return;
    }
    if (state.FillStyle().IsEquivalentColor(parsed_color)) {
      state.SetUnparsedFillColor(isolate, v8_string);
      return;
    }
    state.SetFillColor(parsed_color);
    state.ClearUnparsedFillColor();
    state.ClearResolvedFilter();
    return;
  }
  V8CanvasStyle v8_style;
  if (!ExtractV8CanvasStyle(isolate, value, v8_style, exception_state))
    return;

  UpdateIdentifiabilityStudyBeforeSettingStrokeOrFill(v8_style,
                                                      CanvasOps::kSetFillStyle);

  switch (v8_style.type) {
    case V8CanvasStyleType::kCSSColorValue:
      state.SetFillColor(v8_style.css_color_value);
      break;
    case V8CanvasStyleType::kGradient:
      state.SetFillGradient(v8_style.gradient);
      break;
    case V8CanvasStyleType::kPattern:
      if (!origin_tainted_by_content_ && !v8_style.pattern->OriginClean())
        SetOriginTaintedByContent();
      state.SetFillPattern(v8_style.pattern);
      break;
    case V8CanvasStyleType::kString: {
      Color parsed_color = Color::kTransparent;
      if (ParseColorOrCurrentColor(v8_style.string, parsed_color) ==
          ColorParseResult::kParseFailed) {
        return;
      }
      if (!state.FillStyle().IsEquivalentColor(parsed_color)) {
        state.SetFillColor(parsed_color);
      }
      break;
    }
  }
```

Drawing multiple paths

```
// event 1
ctx.fill(eventPath1);

// event 2
ctx.fill(eventPath2);

// event 3
ctx.fill(eventPath3);
```

JavaScript

```
ctx.fill(eventPath1);
```

C++ (91 LOC)

```
void BaseRenderingContext2D::DrawPathInternal(
    const CanvasPath& path,
    CanvasRenderingContext2DState::PaintType paint_type,
    SkPathFillType fill_type,
    UsePaintCache use_paint_cache) {
  if (path.IsEmpty()) {
    return;
  }

  gfx::RectF bounds(path.BoundingRect());
  if (std::isnan(bounds.x()) || std::isnan(bounds.y()) ||
      std::isnan(bounds.width()) || std::isnan(bounds.height()))
    return;

  if (paint_type == CanvasRenderingContext2DState::kStrokePaintType)
    InflateStrokeRect(bounds);

  if (path.IsLine()) {
    if (paint_type == CanvasRenderingContext2DState::kFillPaintType)
        [[unlikely]] {
      // Filling a line is a no-op.
      // Also, SKCanvas::drawLine() ignores paint type and always strokes.
      return;
    }
    auto line = path.line();
    Draw<OverdrawOp::kNone>(
        [line](cc::PaintCanvas* c,
               const cc::PaintFlags* flags)  // draw lambda
        {
          c->drawLine(SkFloatToScalar(line.start.x()),
                      SkFloatToScalar(line.start.y()),
                      SkFloatToScalar(line.end.x()),
                      SkFloatToScalar(line.end.y()), *flags);
        },
        [](const SkIRect& rect)  // overdraw test lambda
        { return false; },
        bounds, paint_type,
        GetState().HasPattern(paint_type)
            ? CanvasRenderingContext2DState::kNonOpaqueImage
            : CanvasRenderingContext2DState::kNoImage,
        CanvasPerformanceMonitor::DrawType::kPath);
    return;
  }

  if (path.IsArc()) {
    const auto& arc = path.arc();
    const SkScalar x = WebCoreFloatToSkScalar(arc.x);
    const SkScalar y = WebCoreFloatToSkScalar(arc.y);
    const SkScalar radius = WebCoreFloatToSkScalar(arc.radius);
    const SkScalar diameter = radius + radius;
    const SkRect oval =
        SkRect::MakeXYWH(x - radius, y - radius, diameter, diameter);
    const SkScalar start_degrees =
        WebCoreFloatToSkScalar(arc.start_angle_radians * 180 / kPiFloat);
    const SkScalar sweep_degrees =
        WebCoreFloatToSkScalar(arc.sweep_angle_radians * 180 / kPiFloat);
    const bool closed = arc.closed;
    Draw<OverdrawOp::kNone>(
        [oval, start_degrees, sweep_degrees, closed](
            cc::PaintCanvas* c,
            const cc::PaintFlags* flags)  // draw lambda
        {
          cc::PaintFlags arc_paint_flags(*flags);
          arc_paint_flags.setArcClosed(closed);
          c->drawArc(oval, start_degrees, sweep_degrees, arc_paint_flags);
        },
        [](const SkIRect& rect)  // overdraw test lambda
        { return false; },
        bounds, paint_type,
        GetState().HasPattern(paint_type)
            ? CanvasRenderingContext2DState::kNonOpaqueImage
            : CanvasRenderingContext2DState::kNoImage,
        CanvasPerformanceMonitor::DrawType::kPath);
    return;
  }

  SkPath sk_path = path.GetPath().GetSkPath();
  sk_path.setFillType(fill_type);

  Draw<OverdrawOp::kNone>(
      [sk_path, use_paint_cache](cc::PaintCanvas* c,
                                 const cc::PaintFlags* flags)  // draw lambda
      { c->drawPath(sk_path, *flags, use_paint_cache); },
      [](const SkIRect& rect)  // overdraw test lambda
      { return false; },
      bounds, paint_type,
      GetState().HasPattern(paint_type)
          ? CanvasRenderingContext2DState::kNonOpaqueImage
          : CanvasRenderingContext2DState::kNoImage,
      CanvasPerformanceMonitor::DrawType::kPath);
}
```

JavaScript

```
// slow
for (const event of oneThousandEvents) {
    const path = new Path2D();
    path.roundRect(event.x, event.y, event.w, event.h, 10);
    ctx.fillStyle = "pink";
    ctx.fill(path);
}
```

```
// fast
const path = new Path2D();
for (const event of oneThousandEvents) {
    path.roundRect(event.x, event.y, event.w, event.h, 10);
}
ctx.fillStyle = "pink";
ctx.fill(path);
```

C++

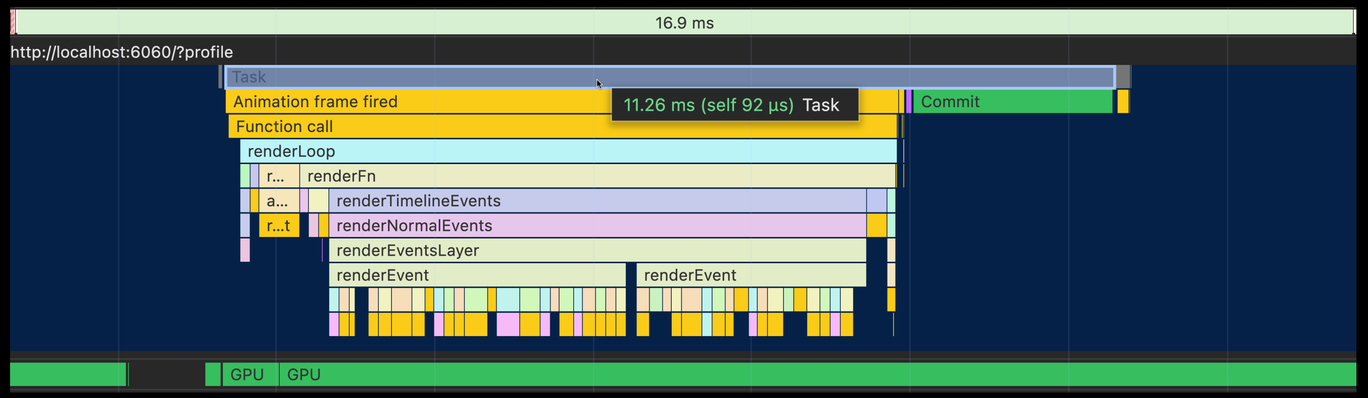
(1 + 133 + 64 + 91) × 1000
= 289,000 LOC

1 + (133 × 1000) + 64 + 91
= 133,156 LOC

289,000 / 133,156 = 2.17x

Slow

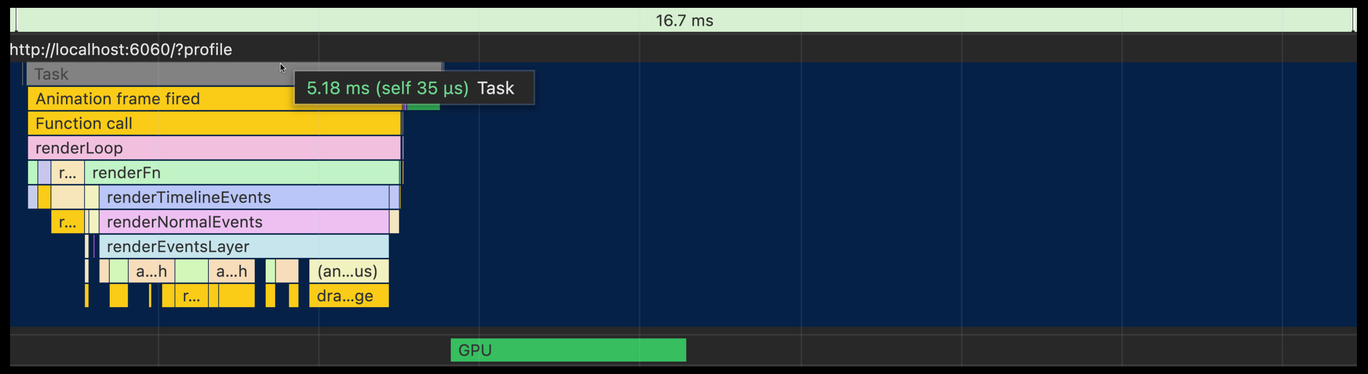
11.26ms



Fast

5.18ms

(2.2x)



Inlining

```ts
const renderEvent = (ctx: CanvasRenderingContext2D, event: TimelineEvent) => {
    {…
    }
};

export const renderEvents = (ctx: CanvasRenderingContext2D, events: TimelineEvent[]) => {
    for (const event of events) {
        renderEvent(ctx, event);
    }
};
```

```
const renderEvent = (ctx: CanvasRenderingContext2D, event: TimelineEvent) => {
    // calculate bounds
    const bounds: Rect = {
        x: calculateX(eventStart, pixelsPerBeat, scroll_px.x),
        y: calculateY(event, voices, scroll_px.y),
        w: calculateWidth(eventDuration, pixelsPerBeat),
        h: kVoiceHeight_px,
    };

    // render background
    {
        ctx.beginPath();
        ctx.roundRect(bounds.x, bounds.y, bounds.w, bounds.h, kEventRadii_px);
        ctx.fillStyle = "pink";
        ctx.fill();
    }

    // [...]
};
```

```
// ---------------------------------- event 1 ----------------------------------
// calculate bounds 1
const bounds1: Rect = {
    x: calculateX(eventStart, pixelsPerBeat, scroll_px.x),
    y: calculateY(event, voices, scroll_px.y),
    w: calculateWidth(eventDuration, pixelsPerBeat),
    h: kVoiceHeight_px,
};

// render background 1
{
    ctx.beginPath();
    ctx.roundRect(bounds1.x, bounds1.y, bounds1.w, bounds1.h, kEventRadii_px);
    ctx.fillStyle = "pink";
    ctx.fill();
}

// ---------------------------------- event 2 ----------------------------------
// calculate bounds 2
const bounds2: Rect = {
    x: calculateX(eventStart, pixelsPerBeat, scroll_px.x),
    y: calculateY(event, voices, scroll_px.y),
    w: calculateWidth(eventDuration, pixelsPerBeat),
    h: kVoiceHeight_px,
};

// render background 2
{
    ctx.beginPath();
    ctx.roundRect(bounds2.x, bounds2.y, bounds2.w, bounds2.h, kEventRadii_px);
    ctx.fillStyle = "pink";
    ctx.fill();
}
```

OUTPUT

```
// ---------------------------------- event 1 ----------------------------------
// calculate bounds 1
const bounds1: Rect = {
    x: calculateX(eventStart, pixelsPerBeat, scroll_px.x),
    y: calculateY(event, voices, scroll_px.y),
    w: calculateWidth(eventDuration, pixelsPerBeat),
    h: kVoiceHeight_px,
};

// render background 1
{
    ctx.beginPath();
    ctx.roundRect(bounds1.x, bounds1.y, bounds1.w, bounds1.h, kEventRadii_px);
    ctx.fillStyle = "pink";
    ctx.fill();
}

// ---------------------------------- event 2 ----------------------------------
// calculate bounds 2
const bounds2: Rect = {
    x: calculateX(eventStart, pixelsPerBeat, scroll_px.x),
    y: calculateY(event, voices, scroll_px.y),
    w: calculateWidth(eventDuration, pixelsPerBeat),
    h: kVoiceHeight_px,
};

// render background 2
{
    ctx.beginPath();
    ctx.roundRect(bounds2.x, bounds2.y, bounds2.w, bounds2.h, kEventRadii_px);
    ctx.fillStyle = "pink";
    ctx.fill();
}
```

OUTPUT

```
// ---------------------------------- event 1 ----------------------------------
{
    ctx.beginPath();
    ctx.roundRect(bounds1.x, bounds1.y, bounds1.w, bounds1.h, kEventRadii_px);
    ctx.fillStyle = "pink";
    ctx.fill();
}

// ---------------------------------- event 2 ----------------------------------
{
    ctx.beginPath();
    ctx.roundRect(bounds2.x, bounds2.y, bounds2.w, bounds2.h, kEventRadii_px);
    ctx.fillStyle = "pink";
    ctx.fill();
}
```

OUTPUT

```
// ---------------------------------- event 1 ----------------------------------
{
    ctx.beginPath();
    ctx.roundRect(bounds1.x, bounds1.y, bounds1.w, bounds1.h, kEventRadii_px);
    ctx.fillStyle = "pink";
    ctx.fill();
}

// ---------------------------------- event 2 ----------------------------------
{
    ctx.beginPath();
    ctx.roundRect(bounds2.x, bounds2.y, bounds2.w, bounds2.h, kEventRadii_px);
    ctx.fillStyle = "pink";
    ctx.fill();
}
```

---------- combined ----------

```
ctx.beginPath();
ctx.roundRect(bounds1.x, bounds1.y, bounds1.w, bounds1.h, kEventRadii_px);
ctx.roundRect(bounds2.x, bounds2.y, bounds2.w, bounds2.h, kEventRadii_px);
ctx.fillStyle = "pink";
ctx.fill();
```

OUTPUT

```
// ------------------------------ combined --------------------------------
{
    ctx.beginPath();
    ctx.roundRect(bounds1.x, bounds1.y, bounds1.w, bounds1.h, kEventRadii_px);
    ctx.roundRect(bounds2.x, bounds2.y, bounds2.w, bounds2.h, kEventRadii_px);
    ctx.fillStyle = "pink";
    ctx.fill();
}
```

```
ctx.beginPath();
for (const event of events) {
    const bounds: Rect = {…
    };
    ctx.roundRect(bounds.x, bounds.y, bounds.w, bounds.h, kEventRadii_px);
}
ctx.fillStyle = "pink";
ctx.fill();
```

OUTPUT

```ts
const renderEvents = (ctx: CanvasRenderingContext2D, events: TimelineEvent[]) => {
    // ---------------------- 1. create paths and view models ----------------------
    const allEventsPath = new Path2D();
    // [...]

    // ---------------------- 2. shared calculations -------------------------------
    const pixelsPerBeat = calculatePixelsPerBeat(zoomLevel);
    // [...]

    // ---------------------- 3. add to paths and view models ----------------------
    for (const event of events) {
        const bounds = createEventBounds(event, pixelsPerBeat);
        allEventsPath.roundRect(bounds.x, bounds.y, bounds.w, bounds.h, kEventRadii_px);
        // [...]
    }

    // ---------------------- 4. rendering -----------------------------------------
    ctx.fillStyle = "pink";
    ctx.fill(allEventsPath);
    // [...]
};
```

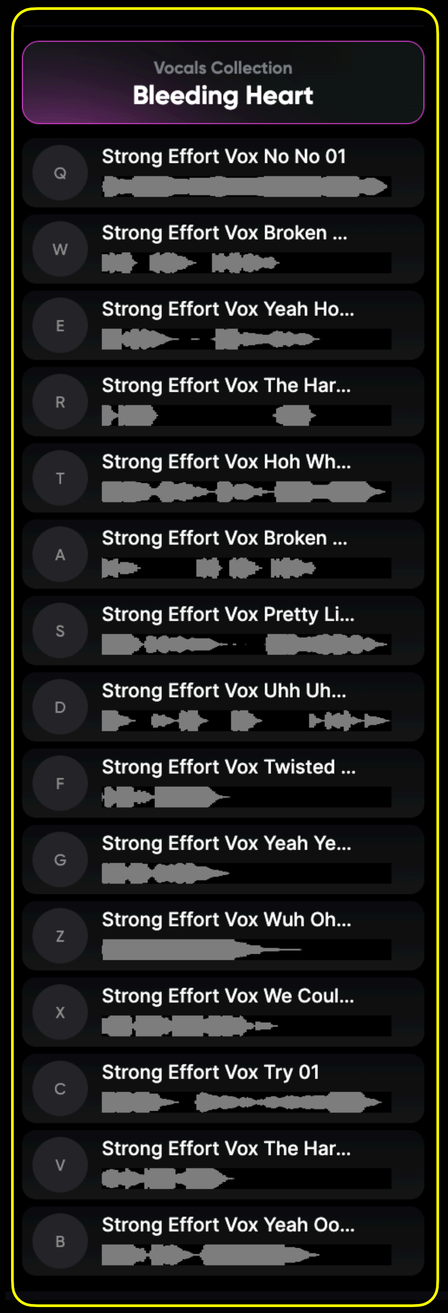
```
// -------------------- 1. create paths and view models --------------------
const allBoundsPath = new Path2D();
const waveformsPath = new Path2D();
const selectionsPath = new Path2D();
const resizeHandlesPath = new Path2D();
const boundsPaths: Record<TrackID, Path2D> = {};
const sampleTextVMs: SampleTextVM[] = [];
const qwertyLabelVMs: Readonly<QwertyLabelVM>[] = [];
```

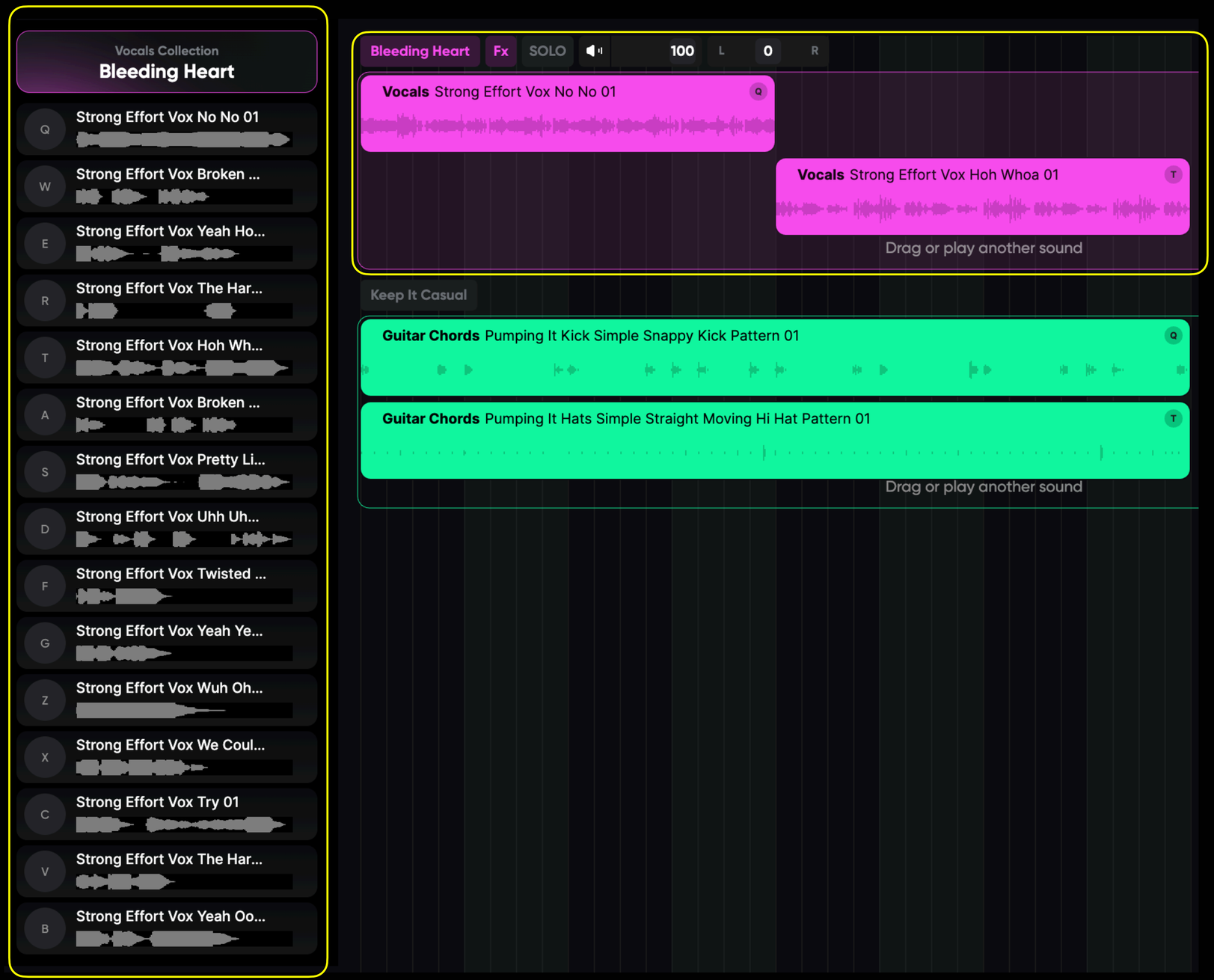
```
// ---------------------- 2. shared calculations ------------------------------
const pixelsPerBeat = calculatePixelsPerBeat(zoomLevel);

// calculate and cache voice y positions
const voiceYs: Record<TrackID, Record<VoiceIndex, number>> = {};
{…
}
```

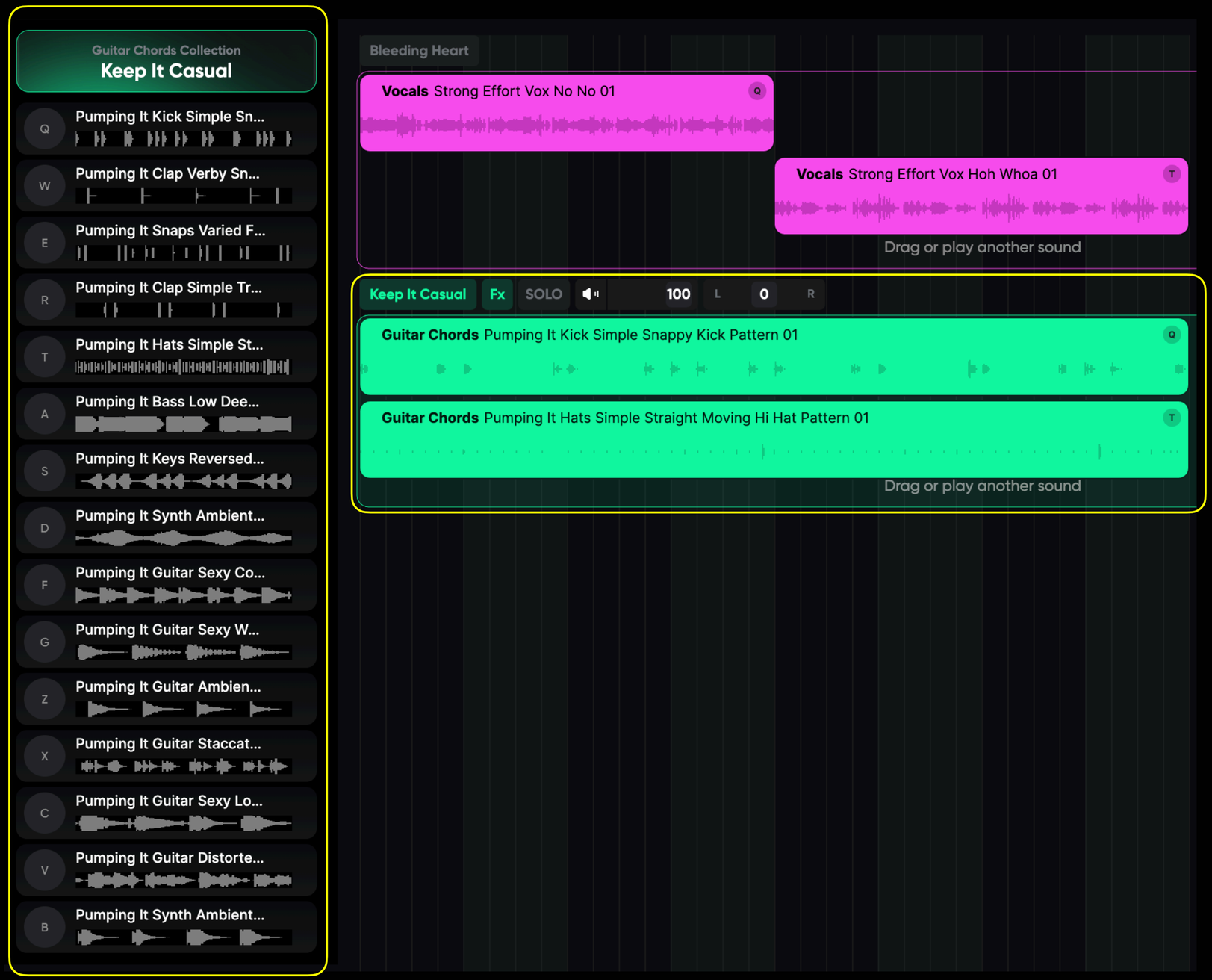
```
// ---------------------- 3. add to paths and view models ----------------------
for (const event of events) {
    const eventY = voiceYs[event.trackId][event.voiceIndex];
    const bounds = createEventRect(event.interval_ppqn, eventY, pixelsPerBeat, scrollX_px);

    // culling
    {…
    }
    // all events bounds path
    {
        allBoundsPath.roundRect(bounds.x, bounds.y, bounds.w, bounds.h, kEventRadii_px);
    }
    // track specific bounds path
    {…
    }
    // text vms
    {…
    }
    // waveforms path
    {…
    }
```

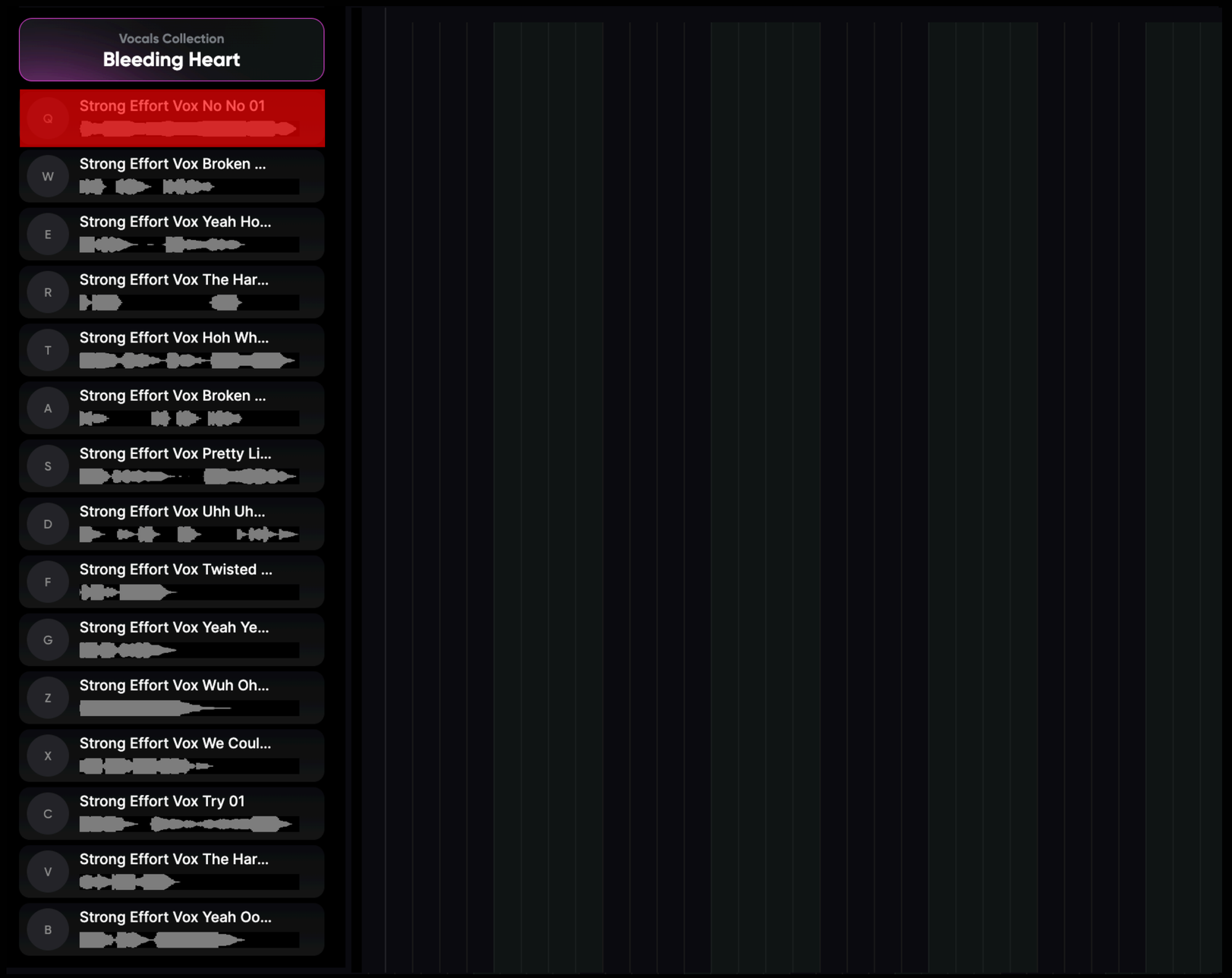
```
// ---------------------- 4. rendering ----------------------------------------
// backgrounds
{…
}

// waveforms
{
    ctx.strokeStyle = waveformColor;
    ctx.lineWidth = kEventWaveformLineWidth_px;
    ctx.stroke(waveformsPath);
}

// selections
{…
}

// resize handles
{…
}
```

1. Canvas
2. **DOM**
3. Layering


Vocals Collection
Bleeding Heart
Q
Strong Effort Vox No No 01
W
Strong Effort Vox Broken ...
E
Strong Effort Vox Yeah Ho...
R
Strong Effort Vox The Har...
T
Strong Effort Vox Hoh Wh...
A
Strong Effort Vox Broken ...
S
Strong Effort Vox Pretty Li...
D
Strong Effort Vox Uhh Uh...
F
Strong Effort Vox Twisted ...
G
Strong Effort Vox Yeah Ye...
Z
Strong Effort Vox Wuh Oh...
X
Strong Effort Vox We Coul...
C
Strong Effort Vox Try 01
V
Strong Effort Vox The Har...
B
Strong Effort Vox Yeah Oo...


Vocals Collection
Bleeding Heart
Q Strong Effort Vox No No 01
W Strong Effort Vox Broken ...
E Strong Effort Vox Yeah Ho...
R Strong Effort Vox The Har...
T Strong Effort Vox Hoh Wh...
A Strong Effort Vox Broken ...
S Strong Effort Vox Pretty Li...
D Strong Effort Vox Uhh Uh...
F Strong Effort Vox Twisted ...
G Strong Effort Vox Yeah Ye...
Z Strong Effort Vox Wuh Oh...
X Strong Effort Vox We Coul...
C Strong Effort Vox Try 01
V Strong Effort Vox The Har...
B Strong Effort Vox Yeah Oo...
Bleeding Heart
Fx
SOLO
100
L
0
R
Vocals Strong Effort Vox No No 01
Q
Vocals Strong Effort Vox Hoh Whoa 01
T
Drag or play another sound
Keep It Casual
Guitar Chords Pumping It Kick Simple Snappy Kick Pattern 01
Q
Guitar Chords Pumping It Hats Simple Straight Moving Hi Hat Pattern 01
T
Drag or play another sound

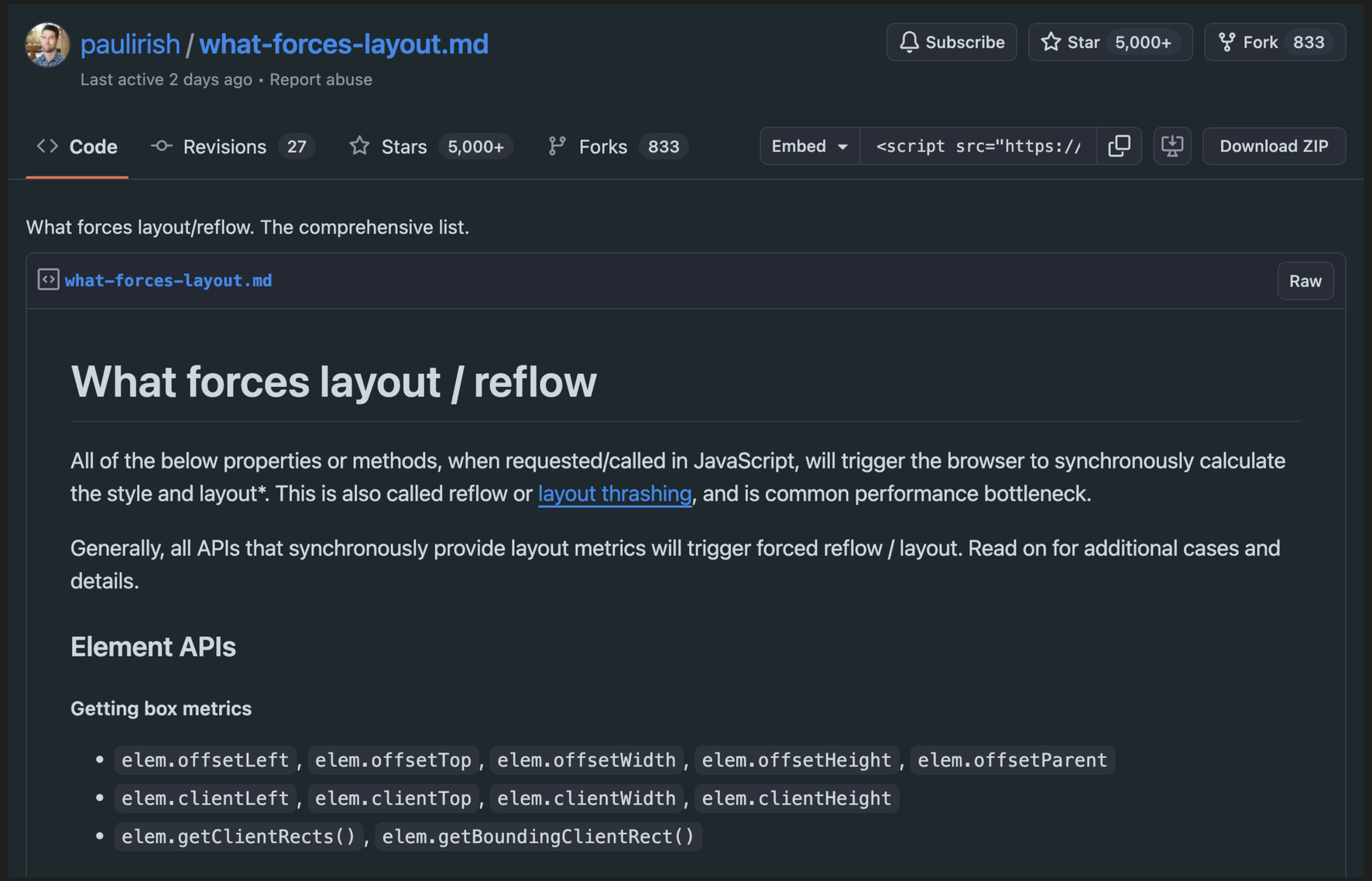

Guitar Chords Collection
Keep It Casual
Q
Pumping It Kick Simple Sn...
W
Pumping It Clap Verby Sn...
E
Pumping It Snaps Varied F...
R
Pumping It Clap Simple Tr...
T
Pumping It Hats Simple St...
A
Pumping It Bass Low Dee...
S
Pumping It Keys Reversed...
D
Pumping It Synth Ambient...
F
Pumping It Guitar Sexy Co...
G
Pumping It Guitar Sexy W...
Z
Pumping It Guitar Ambien...
X
Pumping It Guitar Staccat...
C
Pumping It Guitar Sexy Lo...
V
Pumping It Guitar Distorte...
B
Pumping It Synth Ambient...
Bleeding Heart
Vocals Strong Effort Vox No No 01
Q
Vocals Strong Effort Vox Hoh Whoa 01
T
Drag or play another sound
Keep It Casual
Fx
SOLO
100
L
0
R
Guitar Chords Pumping It Kick Simple Snappy Kick Pattern 01
Q
Guitar Chords Pumping It Hats Simple Straight Moving Hi Hat Pattern 01
T
Drag or play another sound


Vocals Collection
Bleeding Heart
Q
Strong Effort Vox No No 01
W
Strong Effort Vox Broken ...
E
Strong Effort Vox Yeah Ho...
R
Strong Effort Vox The Har...
T
Strong Effort Vox Hoh Wh...
A
Strong Effort Vox Broken ...
S
Strong Effort Vox Pretty Li...
D
Strong Effort Vox Uhh Uh...
F
Strong Effort Vox Twisted ...
G
Strong Effort Vox Yeah Ye...
Z
Strong Effort Vox Wuh Oh...
X
Strong Effort Vox We Coul...
C
Strong Effort Vox Try 01
V
Strong Effort Vox The Har...
B
Strong Effort Vox Yeah Oo...


Vocals Collection
Bleeding Heart
Q
Strong Effort Vox No No 01
W
Strong Effort Vox Broken ...
E
Strong Effort Vox Yeah Ho...
R
Strong Effort Vox The Har...
T
Strong Effort Vox Hoh Wh...
A
Strong Effort Vox Broken ...
S
Strong Effort Vox Pretty Li...
D
Strong Effort Vox Uhh Uh...
F
Strong Effort Vox Twisted ...
G
Strong Effort Vox Yeah Ye...
Z
Strong Effort Vox Wuh Oh...
X
Strong Effort Vox We Coul...
C
Strong Effort Vox Try 01
V
Strong Effort Vox The Har...
B
Strong Effort Vox Yeah Oo...

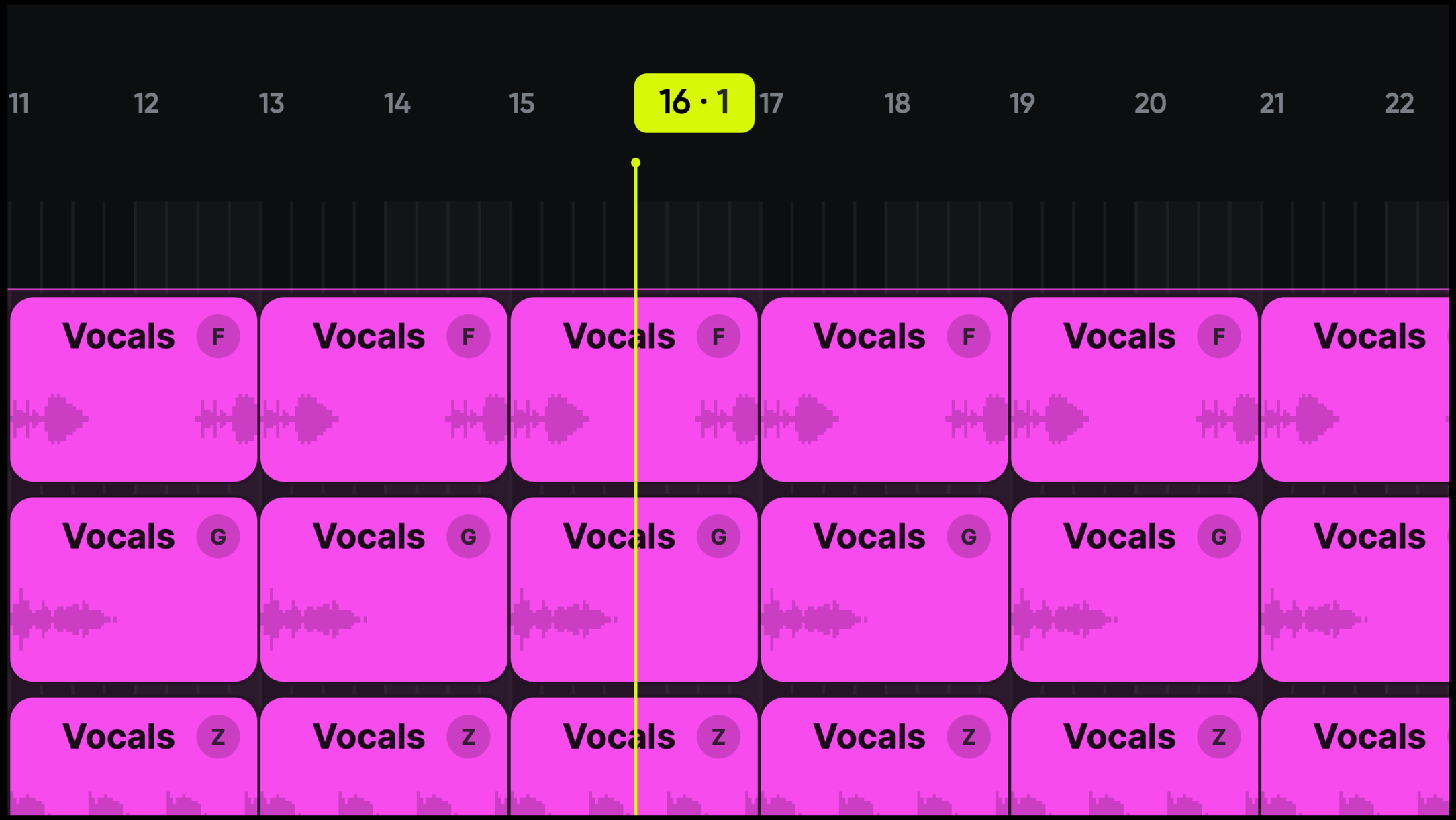
```
// create parent
const listEl = document.createElement("ol");

// create children, add to parent
for (const sample of kit.samples) {
    const sampleEl = createSidebarSampleEl(sample);
    listEl.appendChild(sampleEl);
}

// add all to document
document.body.appendChild(listEl);
```

```
requestAnimationFrame(() => {

    // create parent
    const listEl = document.createElement("ol");

    // create children, add to parent
    for (const sample of kit.samples) {
        const sampleEl = createSidebarSampleEl(sample);
        listEl.appendChild(sampleEl);
    }

    // add all to document
    document.body.appendChild(listEl);

});
```

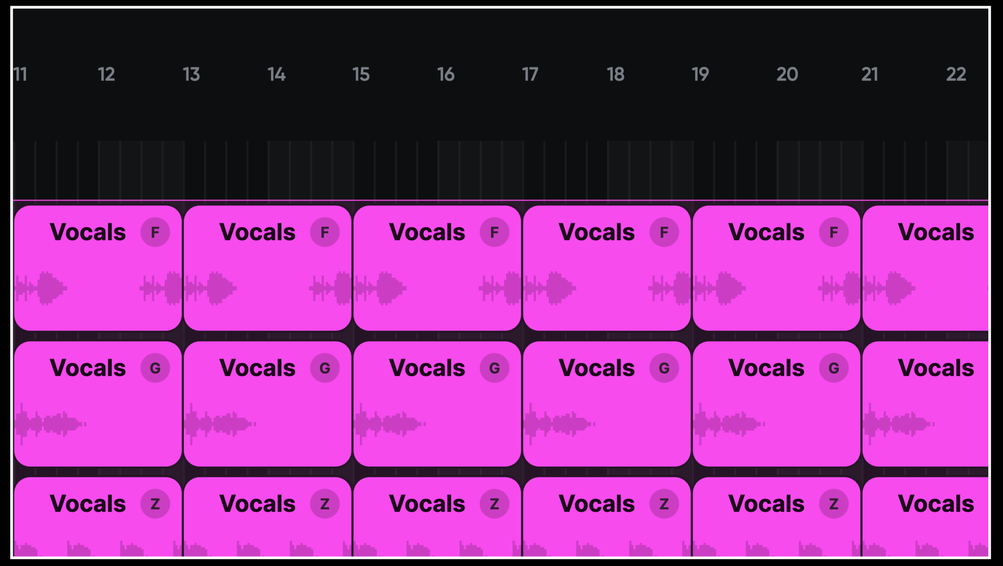


https://gist.github.com/paulirish/5d52fb081b3570c81e3a

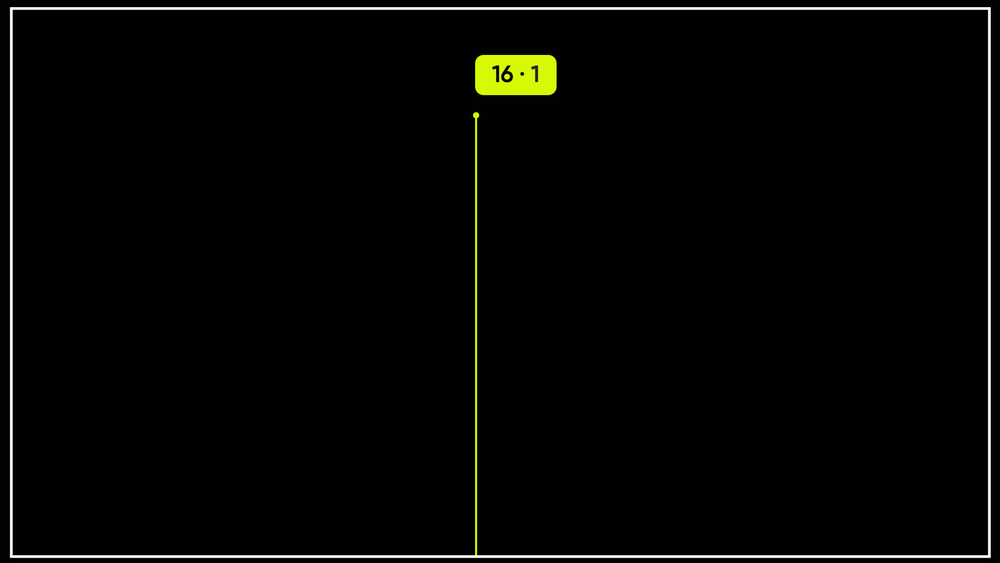
1. Canvas
2. DOM
3. **Layering**


11
12
13
14
15
16 · 1
17
18
19
20
21
22
Vocals F
Vocals F
Vocals F
Vocals F
Vocals F
Vocals
Vocals G
Vocals G
Vocals G
Vocals G
Vocals G
Vocals
Vocals Z
Vocals Z
Vocals Z
Vocals Z
Vocals Z
Vocals

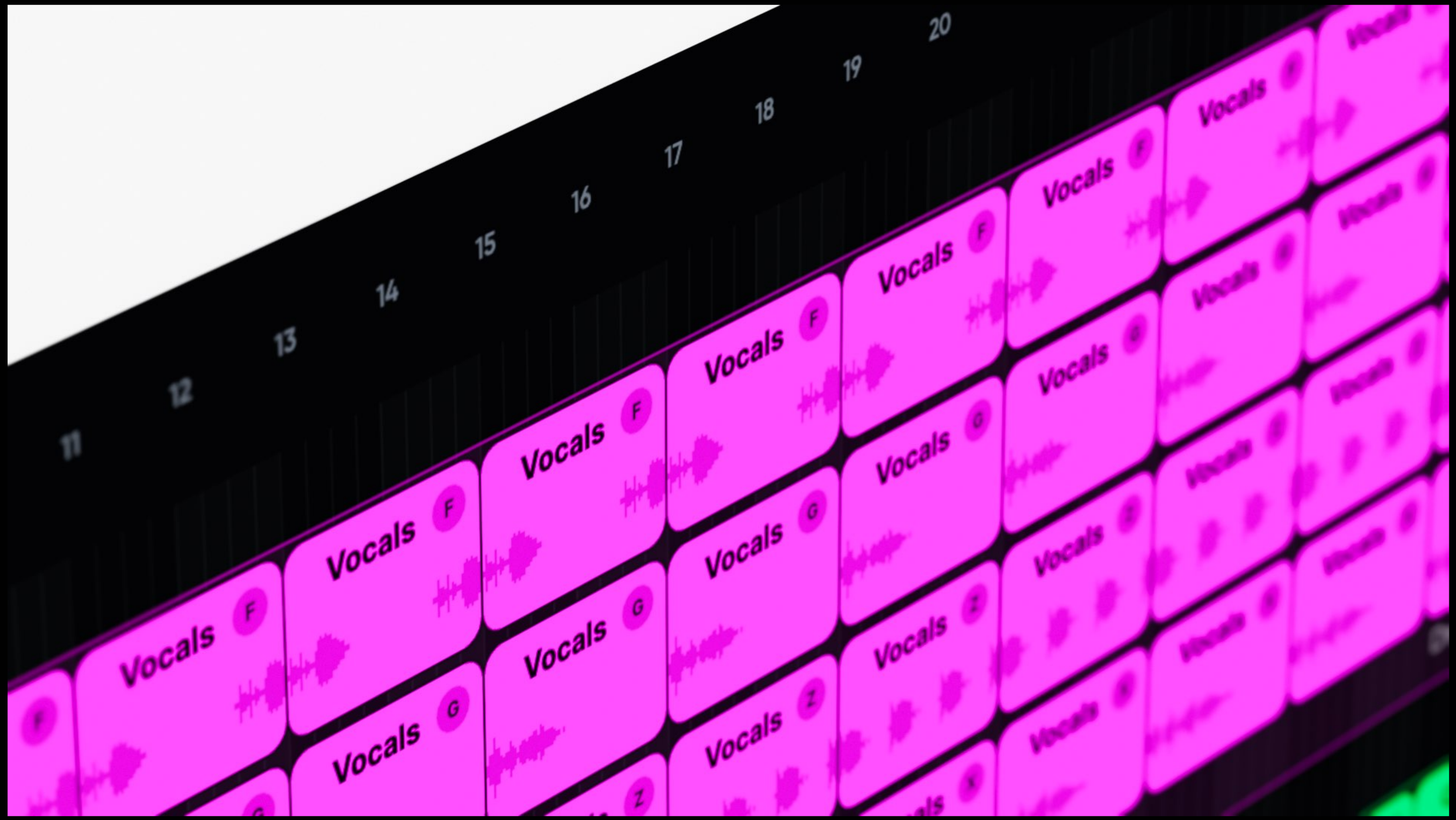
Canvas 1



Canvas 2



```
<canvas style="z-index: 1;"></canvas> <!-- Events   -->
<canvas style="z-index: 2;"></canvas> <!-- Playhead -->
```


11
12
13
14
15
16
17
18
19
20
Vocals
F
G
Z


11
12
13
14
15
16
17
18
19
20
16 · 1
Vocals
F
G
Z


3
4
5
Bass
Blue Orchid Bass Intro 01
Q

In conclusion...

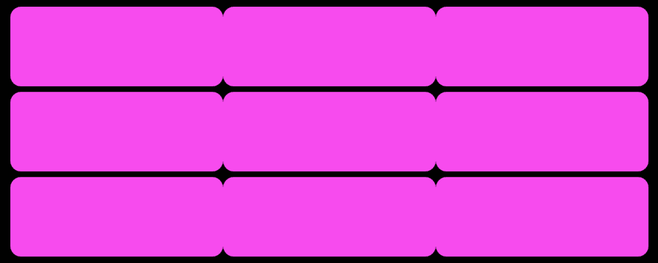
Batch

It is very unlikely in any given situation that one thing exists in a vacuum.

Mike Acton
CppCon 2014: "Data-Oriented Design and C++" 37m 30s
https://www.youtube.com/watch?v=rX0ItVEVjHc

Consider the context

Batch when visual properties are the same...



Batch when changes happen at the same time

Separate entirely where appropriate

Slides and book mailing list

arthurcarabott.com/adc-2024

Contact

arthur@arthurcarabott.com