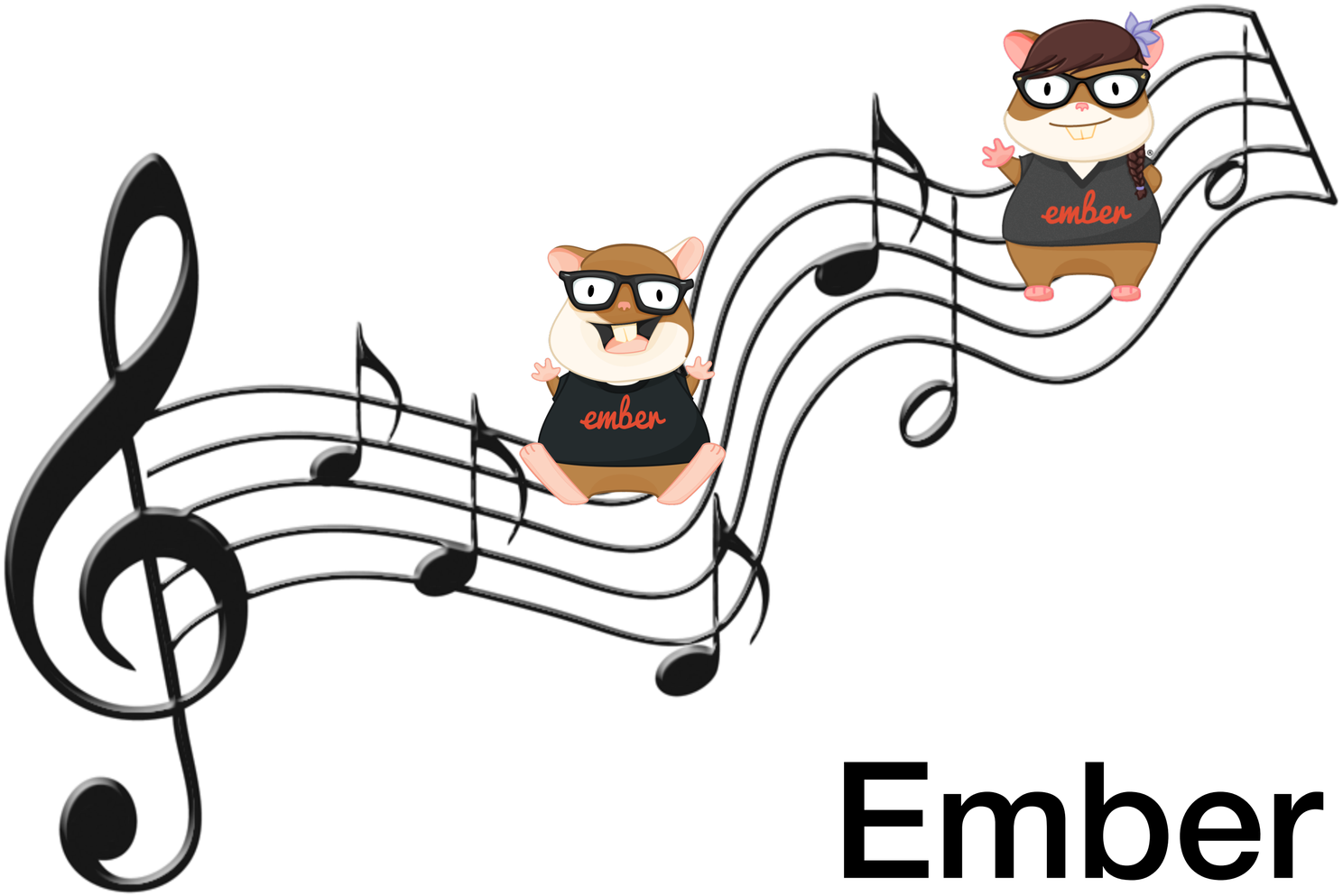


Ember as Song

James C. Davis

An Idea

- Brainstorming session
- Melanie Sumner
- Compose a song alongside building an Ember app

The Idea Grows

- How to best compare building an app with composing a song
- Map elements of song to concepts in Ember
- An Ember app that IS a song

Programming and Songwriting

- Both creative endeavors
- Both are complex
- Share many fundamental concepts
  - Patterns
  - Loops
  - Conditionals
  - Problem solving
  - Bugfixing

What’s in a song?

- Sections
  - Intro
  - Verse
  - Chorus
  - Bridge

What are sections composed of?

- Instruments
- Parts
- Notes
- Measures
- Phrases

Mapping to Ember

- Sections -> Routes
- Instruments -> Services
- Parts -> Components
- Notes -> Contextual Components

Section -> Route

- A location in the song
- Composed of instruments playing parts together
- The musical equivalent of a “page”

Instrument -> Service

- Instruments are global
- Maintain state when moving through sections
  - Keep playing
  - Volume
  - Effects

Part -> Component

- Made up of notes for one instrument
- Often a pattern that repeats within a section
- Often re-used across multiple sections

Note -> Contextual Component

- Belongs to a part
- Needs context
  - Which instrument
  - When to play relative to other notes

Other Globals

- Tempo
- Master Volume
- Playing state (start, pause, stop)
- Where we are in the timeline
- Could be handled by a single or multiple services

Concept -> Implementation

- Web Audio API - https://www.w3.org/TR/webaudio/
  - Extremely powerful, but very low level
- Tone.js - https://tonejs.github.io
  - Built on top of Web Audio API
  - Primitives for making music (not just sounds)

Hello World

- Create one instrument: piano
- Play a single note: middle C

Service: piano

ember g service piano

```
import Service from '@ember/service';
import SampleLibrary from 'ember-as-song/lib/sample-library';

export default class PianoService extends Service {
  name = 'Piano';

  inst = SampleLibrary.load({
    instruments: 'piano'
  }).toMaster();
}
```

Instrument Component

```
<h3 local-class='title'>{{@instrument.name}}</h3>

<div local-class='container {{if @parallel 'parallel'}}'>
  {{yield (hash
    part=(component
      'instrument-part'
      instrument=@instrument
      volume=@volume
      humanize=@humanize
    )
  )}}
</div>
```

Part Component

- Responsible for
  - Connecting notes to an instrument
  - Scheduling notes to play
  - Scheduling draws
  - Looping

Part Component

```
<div {{did-insert this.initPart}} local-class='part'>
  <div local-class='container'>
    {{yield (hash
      note=(component 'part-note'
        addNote=this.addNote
        activeNote=this.activeNote
      )
    )}}
  </div>
</div>
```

Note Component

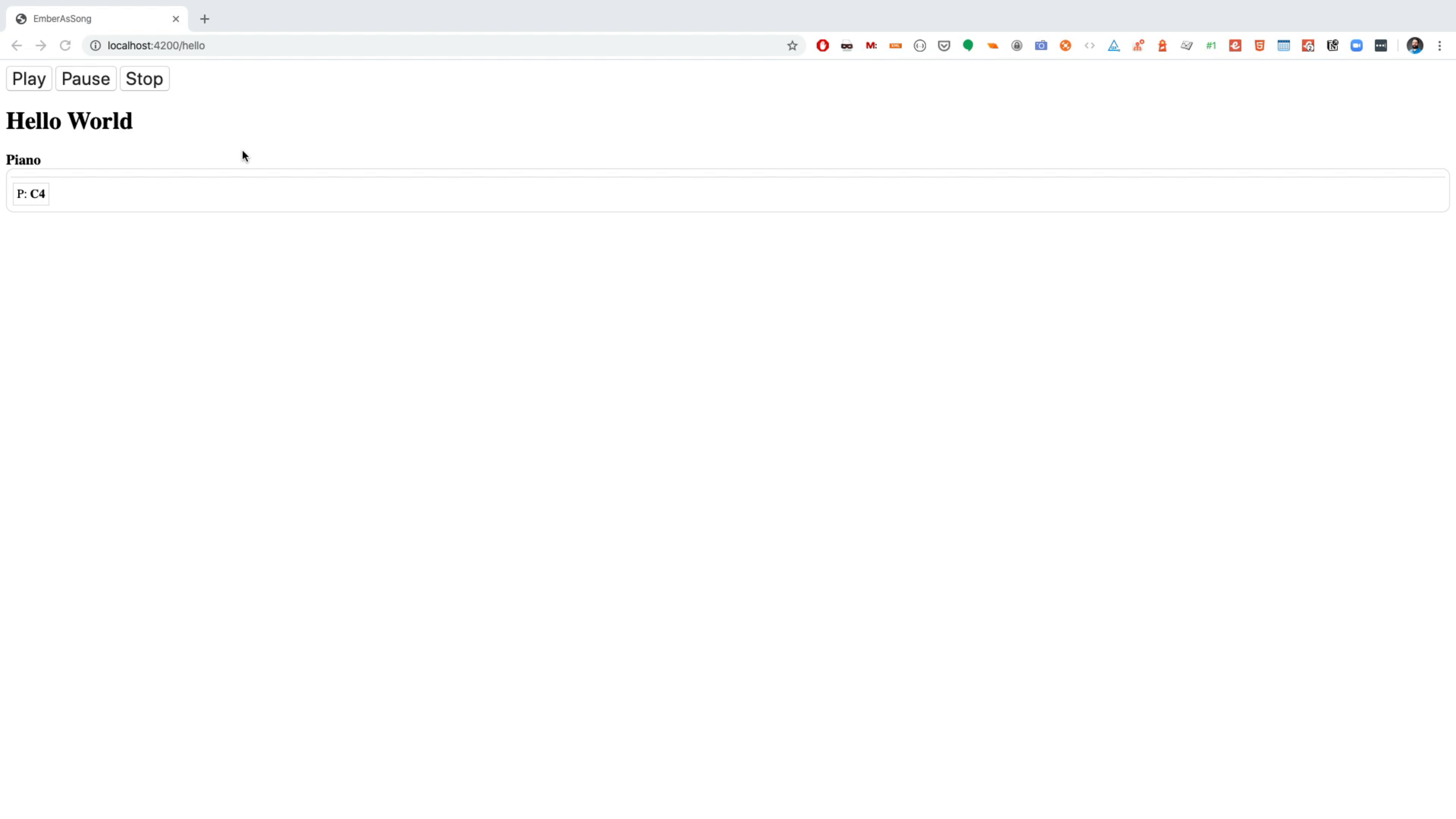
```
<div local-class='note {{if this.active 'active'}}'>
  <span>T: <strong>{{this.time}}</strong></span>
  <span>P: <strong>{{this.pitch}}</strong></span>
</div>
```

```
.active {
  background-color: #fa4;
}
```

```
import Component from '@glimmer/component';
import { tracked } from '@glimmer/tracking';

export default class PartAddNoteComponent extends Component {
  @tracked id;

  constructor(owner, args) {
    super(owner, args);
    const { pitch, duration, time = '0', velocity } = this.args;
    this.id = args.addNote({ pitch, duration, time, velocity });
  }

  get active() {
    return this.args.activeNote === this.id;
  }
}
```

Play it!

```
<Instrument @instrument={{this.piano}} as |i|>
  <i.part as |p|>
    <p.note @pitch='C4' />
  </i.part>
</Instrument>
```

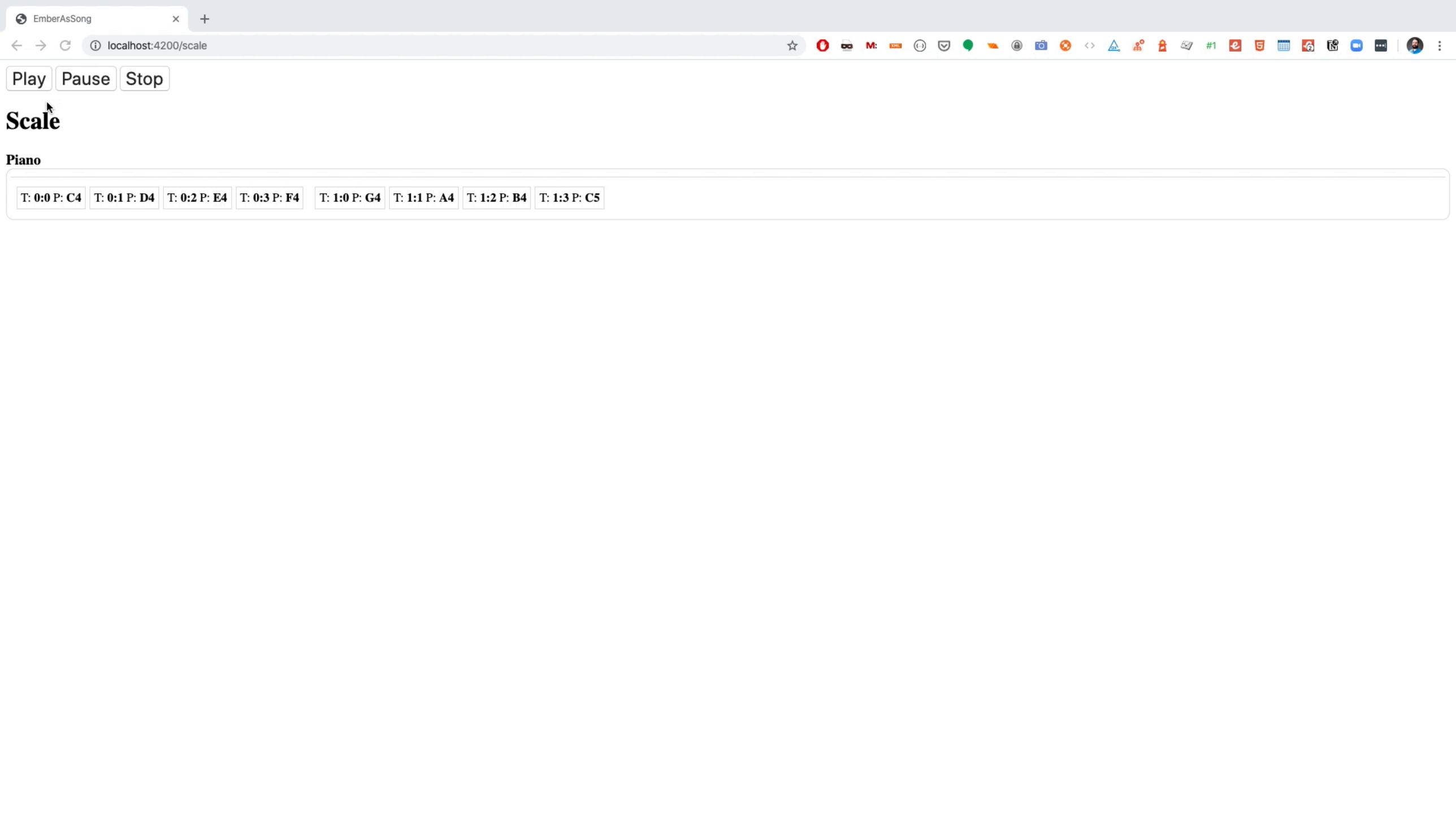

EmberAsSong
localhost:4200/hello
Play
Pause
Stop
Hello World
Piano
P: C4

Scale

- String of notes, one after another
- Ascending (or descending) in pitch
- Hitting all the white keys (for a C scale)

Scale

```
<Instrument @instrument={{this.piano}} as |i|>
  <i.part as |p|>
    <Measure>
      <p.note @time='0:0' @pitch='C4' />
      <p.note @time='0:1' @pitch='D4' />
      <p.note @time='0:2' @pitch='E4' />
      <p.note @time='0:3' @pitch='F4' />
    </Measure>

    <Measure>
      <p.note @time='1:0' @pitch='G4' />
      <p.note @time='1:1' @pitch='A4' />
      <p.note @time='1:2' @pitch='B4' />
      <p.note @time='1:3' @pitch='C5' />
    </Measure>
  </i.part>
</Instrument>
```


EmberAsSong
localhost:4200/scale
Play
Pause
Stop
Scale
Piano
T: 0:0 P: C4
T: 0:1 P: D4
T: 0:2 P: E4
T: 0:3 P: F4
T: 1:0 P: G4
T: 1:1 P: A4
T: 1:2 P: B4
T: 1:3 P: C5

How does this work?

```
<div {{did-insert this.initPart}} local-class='part'>
  <div local-class='container'>
    {{yield (hash
      note=(component 'part-note'
        addNote=this.addNote
        activeNote=this.activeNote
      )
    )}}
  </div>
</div>
```

```
export default class PartAddNoteComponent extends Component {
  @tracked id;

  constructor(owner, args) {
    super(owner, args);
    const { pitch, duration, time = '0', velocity } = this;
    this.id = args.addNote({ pitch, duration, time, velocity });
  }
}
```

```
export default class PartComponent extends Component {
  notes = [];

  @action
  addNote(note) {
    const id = this.notes.length;
    this.notes.push({ id, ...note });
    return id;
  }

  @action
  triggerSynth(time, note) {
    const { pitch, duration = '4n', velocity } = note;
    const { inst } = this.instrument;
    inst.triggerAttackRelease(pitch, duration, time, velocity);
  }

  @action
  initPart() {
    this.part = new Part(this.triggerSynth, this.notes);
  }
}
```

Visualization

```
<div local-class='note {{if this.active 'active'}}'>
  <span>T: <strong>{{this.time}}</strong></span>
  <span>P: <strong>{{this.pitch}}</strong></span>
</div>
```

```
export default class PartComponent extends Component {
  @tracked activeNote = -1;

  @action
  triggerSynth(time, note) {
    const { pitch, duration = '4n', velocity } = note;
    const { inst } = this.instrument;
    inst.triggerAttackRelease(pitch, duration, time, velocity);
    Draw.schedule(() => {
      this.activeNote = note.id;
    }, time);
    Draw.schedule(() => {
      this.activeNote = -1;
    }, time + Time(duration).toSeconds());
  }
```

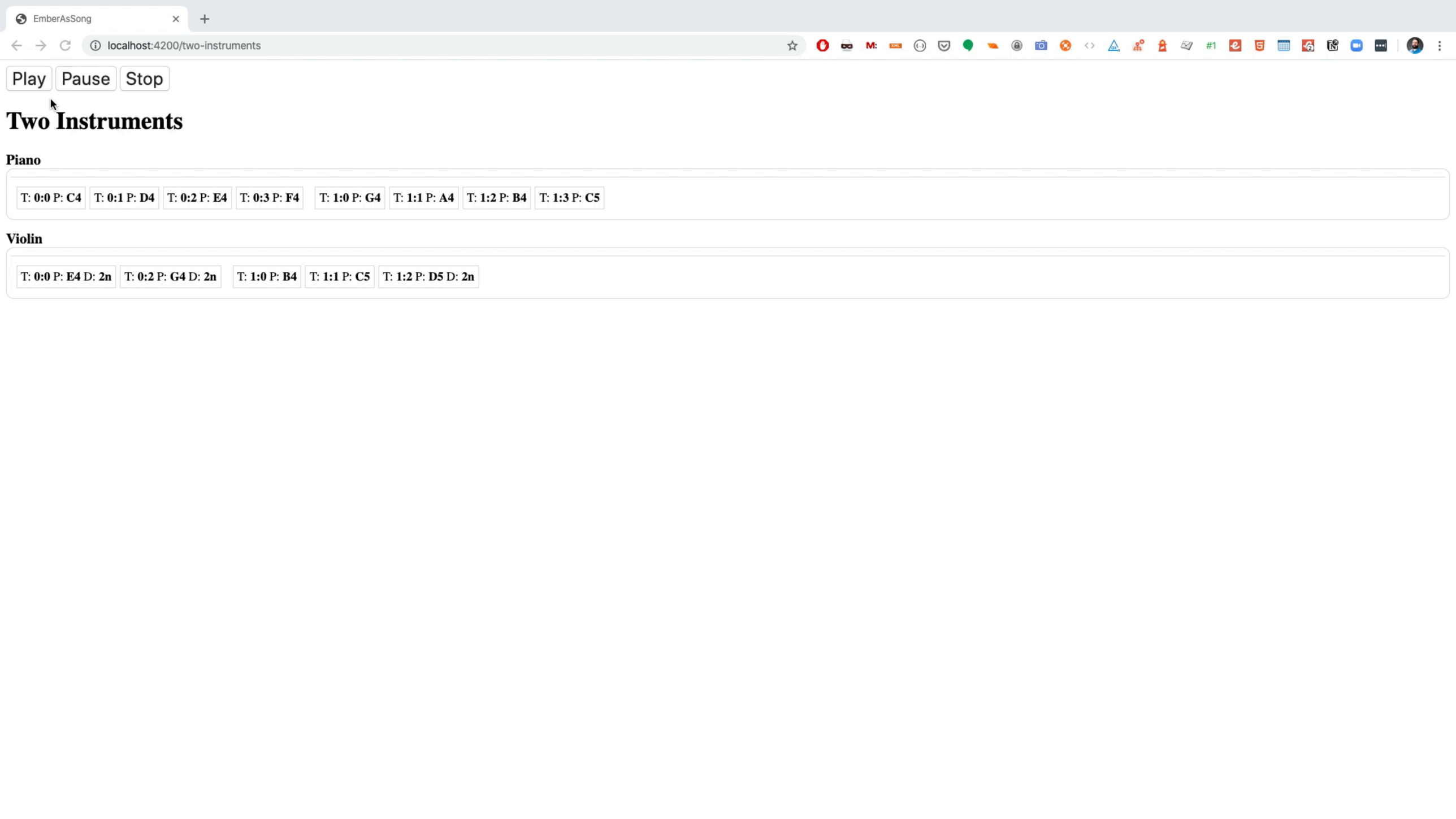
```
  get active() {
    return this.args.activeNote === this.id;
  }
```

```
.active {
  background-color: #fa4;
}
```

Two Instruments

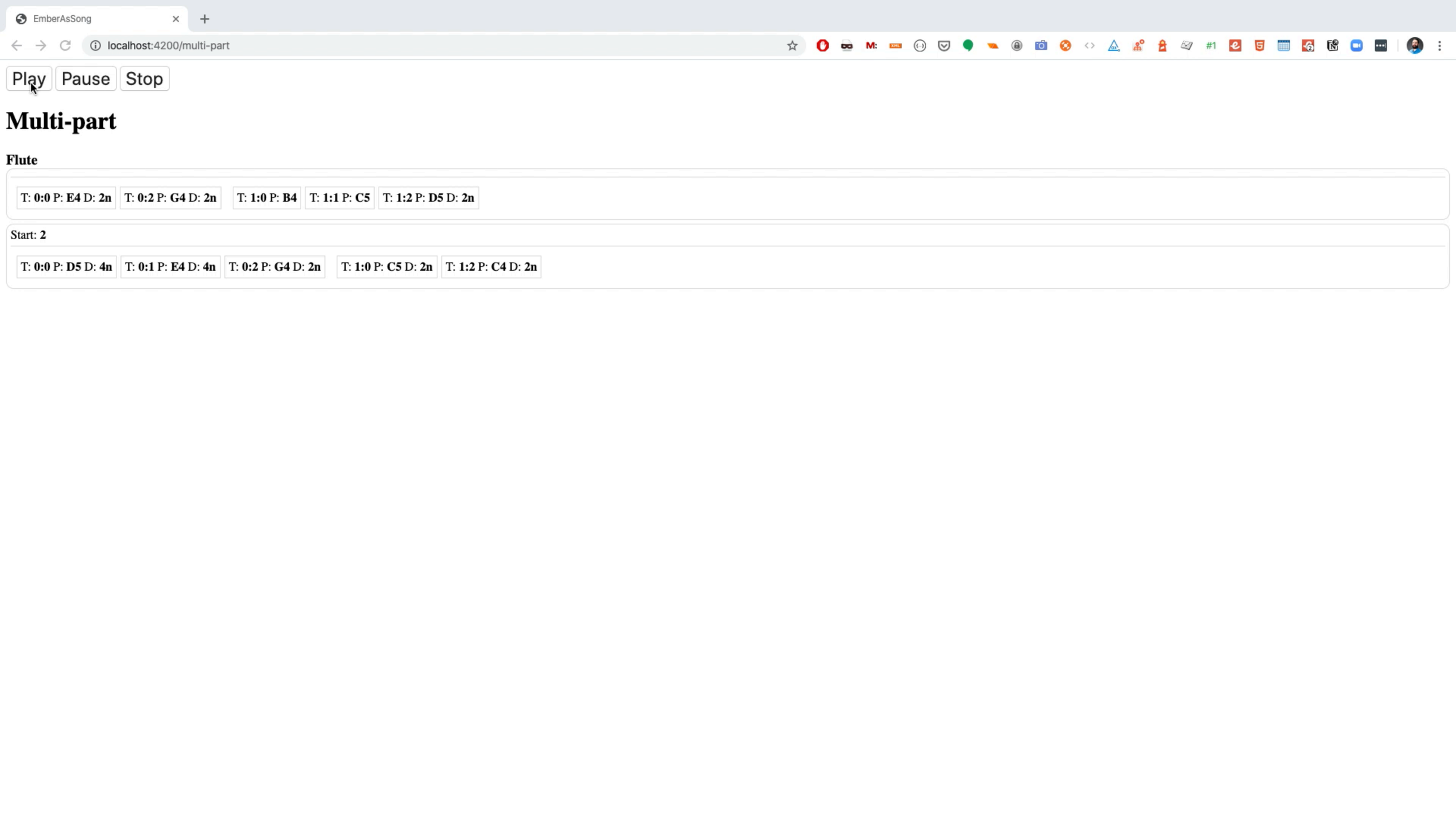
```
<Instrument @instrument={{this.piano}} as |i|>
  <i.part as |p|>
    <Measure>
      <p.note @time='0:0' @pitch='C4' />
      <p.note @time='0:1' @pitch='D4' />
      <p.note @time='0:2' @pitch='E4' />
      <p.note @time='0:3' @pitch='F4' />
    </Measure>

    <Measure>
      <p.note @time='1:0' @pitch='G4' />
      <p.note @time='1:1' @pitch='A4' />
      <p.note @time='1:2' @pitch='B4' />
      <p.note @time='1:3' @pitch='C5' />
    </Measure>
  </i.part>
</Instrument>
```

```
<Instrument @instrument={{this.violin}} as |i|>
  <i.part as |p|>
    <Measure>
      <p.note @time='0:0' @pitch='E4' @duration='2n' />
      <p.note @time='0:2' @pitch='G4' @duration='2n' />
    </Measure>

    <Measure>
      <p.note @time='1:0' @pitch='B4' />
      <p.note @time='1:1' @pitch='C5' />
      <p.note @time='1:2' @pitch='D5' @duration='2n' />
    </Measure>
  </i.part>
</Instrument>
```


EmberAsSong
localhost:4200/two-instruments
Play
Pause
Stop
Two Instruments
Piano
T: 0:0 P: C4
T: 0:1 P: D4
T: 0:2 P: E4
T: 0:3 P: F4
T: 1:0 P: G4
T: 1:1 P: A4
T: 1:2 P: B4
T: 1:3 P: C5
Violin
T: 0:0 P: E4 D: 2n
T: 0:2 P: G4 D: 2n
T: 1:0 P: B4
T: 1:1 P: C5
T: 1:2 P: D5 D: 2n

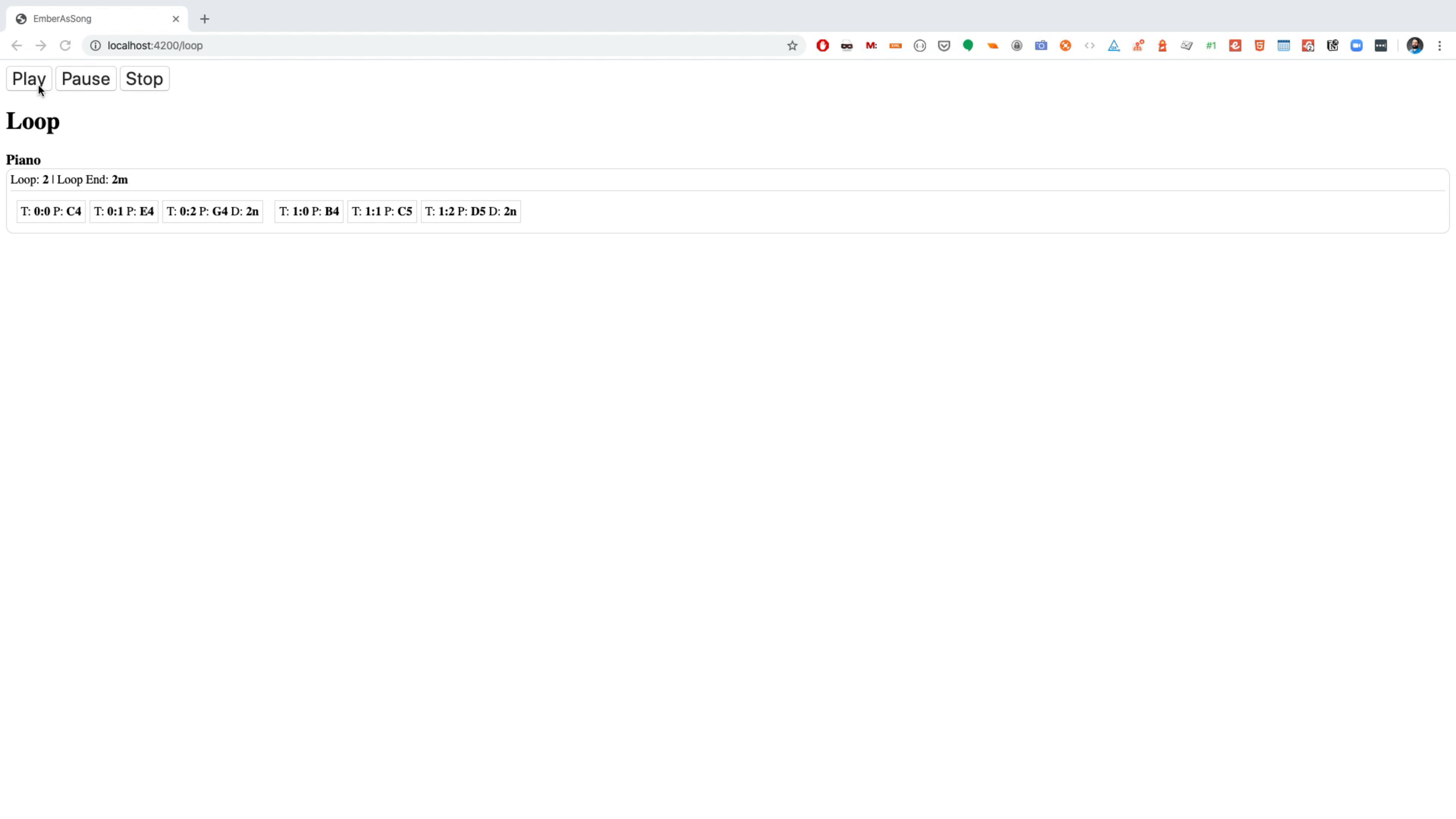
Multi-part

```
<Instrument @instrument={{this.flute}} as |i|>
  <i.part @start={{0}} as |p|>
    <Measure>
      <p.note @time='0:0' @pitch='E4' @duration='2n' />
      <p.note @time='0:2' @pitch='G4' @duration='2n' />
    </Measure>

    <Measure>
      <p.note @time='1:0' @pitch='B4' />
      <p.note @time='1:1' @pitch='C5' />
      <p.note @time='1:2' @pitch='D5' @duration='2n' />
    </Measure>
  </i.part>

  <i.part @start={{2}} as |p|>
    <Measure>
      <p.note @time='0:0' @pitch='D5' @duration='4n' />
      <p.note @time='0:1' @pitch='E4' @duration='4n' />
      <p.note @time='0:2' @pitch='G4' @duration='2n' />
    </Measure>

    <Measure>
      <p.note @time='1:0' @pitch='C5' @duration='2n' />
      <p.note @time='1:2' @pitch='C4' @duration='2n' />
    </Measure>
  </i.part>
</Instrument>
```


EmberAsSong
localhost:4200/multi-part
Play
Pause
Stop
Multi-part
Flute
T: 0:0 P: E4 D: 2n
T: 0:2 P: G4 D: 2n
T: 1:0 P: B4
T: 1:1 P: C5
T: 1:2 P: D5 D: 2n
Start: 2
T: 0:0 P: D5 D: 4n
T: 0:1 P: E4 D: 4n
T: 0:2 P: G4 D: 2n
T: 1:0 P: C5 D: 2n
T: 1:2 P: C4 D: 2n

Loops

```
<Instrument @instrument={{this.piano}} as |i|>
  <i.part @loop={{2}} @loopEnd='2m' as |p|>
    <Measure>
      <p.note @time='0:0' @pitch='C4' />
      <p.note @time='0:1' @pitch='E4' />
      <p.note @time='0:2' @pitch='G4' @duration='2n' />
    </Measure>

    <Measure>
      <p.note @time='1:0' @pitch='B4' />
      <p.note @time='1:1' @pitch='C5' />
      <p.note @time='1:2' @pitch='D5' @duration='2n' />
    </Measure>
  </i.part>
</Instrument>
```


EmberAsSong
localhost:4200/loop
Play
Pause
Stop
Loop
Piano
Loop: 2 | Loop End: 2m
T: 0:0 P: C4
T: 0:1 P: E4
T: 0:2 P: G4 D: 2n
T: 1:0 P: B4
T: 1:1 P: C5
T: 1:2 P: D5 D: 2n

Drum Kit

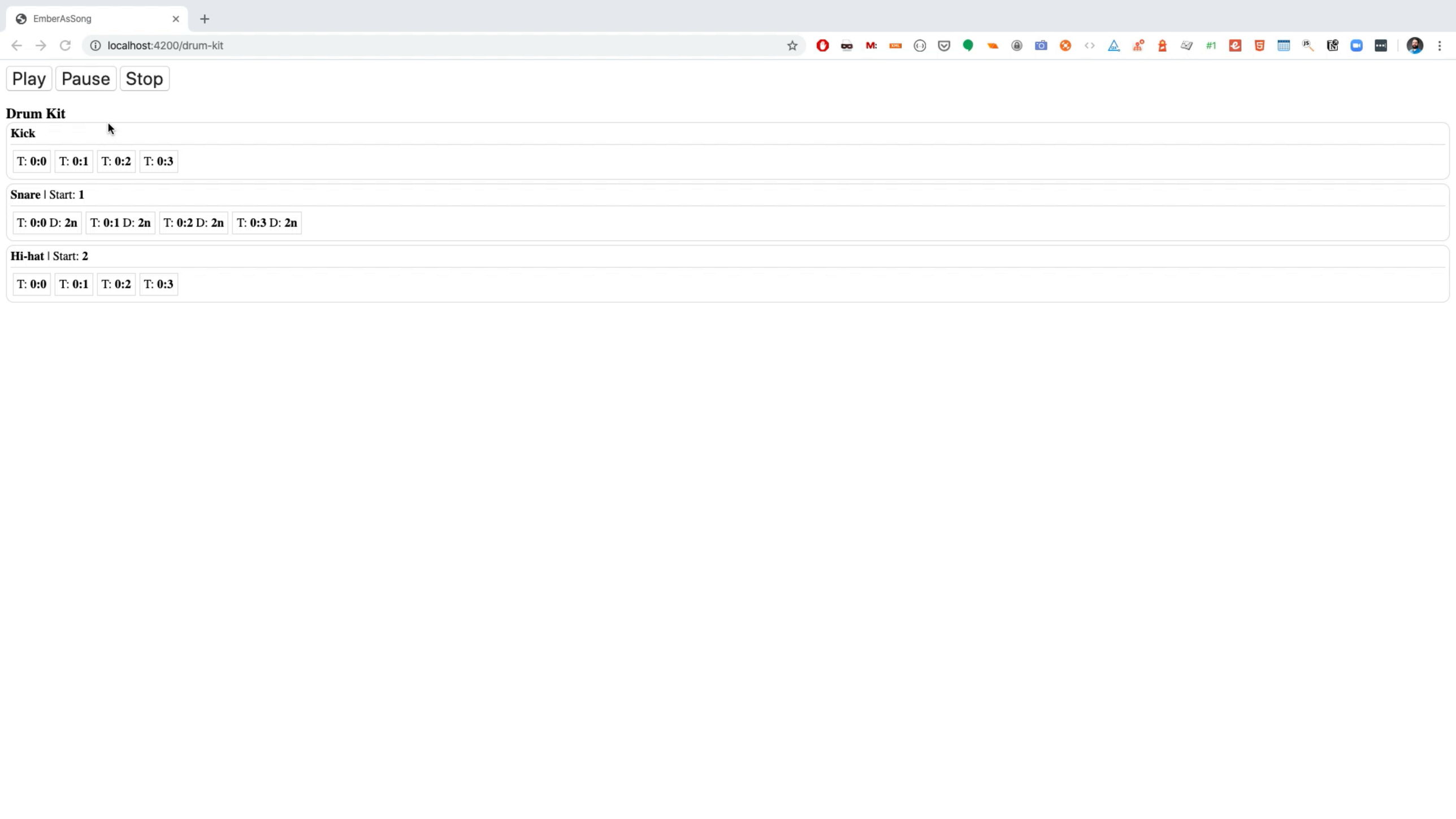
```
export default class DrumsService extends Service {
  name = 'Drum Kit';

  kick = {
    name: 'Kick',
    defaultPitch: 'C1',
    inst: new MembraneSynth({
      pitchDecay: 0.07,
      octaves: 5,
      oscillator: {
        type: 'sine'
      },
      envelope: {
        attack: 0.001,
        decay: 0.2,
        sustain: 0.002,
        release: 1,
        attackCurve: 'exponential'
      }
    }).toMaster(),
  };
```

```
export default class DrumsService extends Service {
  name = 'Drum Kit';

  snare = {
    name: 'Snare',
    defaultPitch: 'C4',
    inst: SampleLibrary.load({
      instruments: 'snare'
    }).toMaster(),
  };
```

```
export default class DrumsService extends Service {
  name = 'Drum Kit';

  hihatPan = new PanVol({
    pan: 0.35
  }).toMaster();

  hihat = {
    name: 'Hi-hat',
    inst: new MetalSynth({
      frequency: 200,
      envelope: {
        attack: 0.008,
        decay: 0.052,
        release: 0.002
      },
      harmonicity: 5.1,
      modulationindex: 32,
      resonance: 3000,
      octaves: 1.5
    }).connect(this.hihatPan),
  }
```

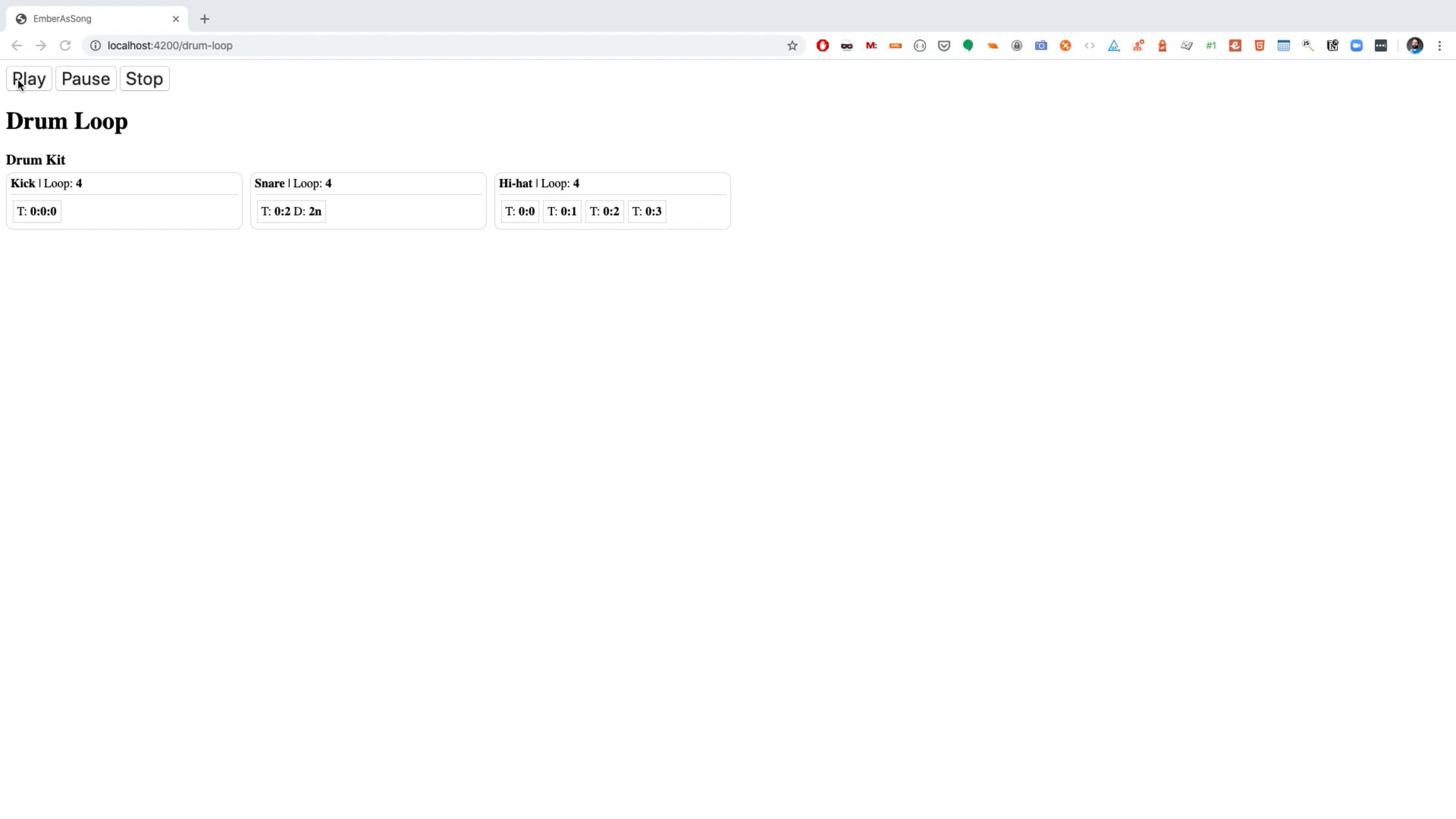
Drum Kit

```
<Instrument @instrument={{this.drums}} as |i|>
  <i.part @subInstrument='kick' as |p|>
    <p.note @t='0:0' />
    <p.note @t='0:1' />
    <p.note @t='0:2' />
    <p.note @t='0:3' />
  </i.part>

  <i.part @subInstrument='snare' @volume={{-3}} @start={{1}} as |p|>
    <p.note @t='0:0' @d='2n' />
    <p.note @t='0:1' @d='2n' />
    <p.note @t='0:2' @d='2n' />
    <p.note @t='0:3' @d='2n' />
  </i.part>

  <i.part @subInstrument='hihat' @volume={{-20}} @humanize={{true}} @start={{2}} as |p|>
    <p.note @t='0:0' />
    <p.note @t='0:1' />
    <p.note @t='0:2' />
    <p.note @t='0:3' />
  </i.part>
</Instrument>
```


EmberAsSong
localhost:4200/drum-kit
Play
Pause
Stop
Drum Kit
Kick
T: 0:0
T: 0:1
T: 0:2
T: 0:3
Snare | Start: 1
T: 0:0 D: 2n
T: 0:1 D: 2n
T: 0:2 D: 2n
T: 0:3 D: 2n
Hi-hat | Start: 2
T: 0:0
T: 0:1
T: 0:2
T: 0:3

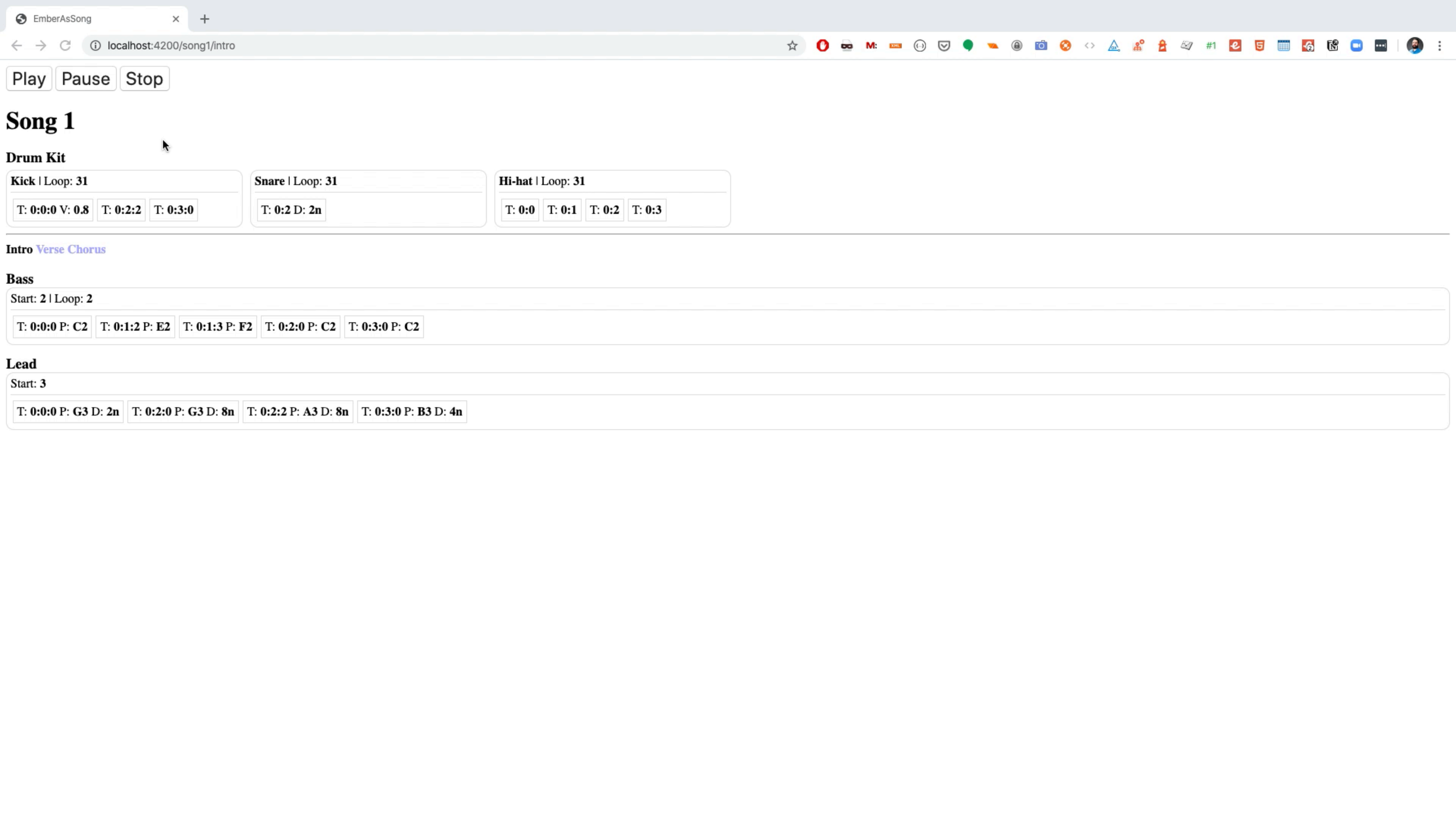
Drum Loop

```
<Instrument @instrument={{this.drums}} @loop={{4}} @parallel={{true}} as |i|>
  <i.part @subInstrument='kick' as |p|>
    <p.note @t='0:0:0' />
  </i.part>

  <i.part @subInstrument='snare' @volume={{-3}} as |p|>
    <p.note @t='0:2' @d='2n' />
  </i.part>

  <i.part @subInstrument='hihat' @volume={{-20}} @humanize={{true}} as |p|>
    <p.note @t='0:0' />
    <p.note @t='0:1' />
    <p.note @t='0:2' />
    <p.note @t='0:3' />
  </i.part>
</Instrument>
```


EmberAsSong
localhost:4200/drum-loop
Play
Pause
Stop
Drum Loop
Drum Kit
Kick | Loop: 4
T: 0:0:0
Snare | Loop: 4
T: 0:2 D: 2n
Hi-hat | Loop: 4
T: 0:0
T: 0:1
T: 0:2
T: 0:3

Putting it all Together

- Composed a song with Ember!
- Routes and controllers for: intro, verse, chorus
- Auto-advances among the sections
- Routes handle timing and advancing
- Controllers inject instruments (services)
- Parent route for drums


EmberAsSong
localhost:4200/song1/intro
Play
Pause
Stop
Song 1
Drum Kit
Kick | Loop: 31
T: 0:0:0 V: 0.8
T: 0:2:2
T: 0:3:0
Snare | Loop: 31
T: 0:2 D: 2n
Hi-hat | Loop: 31
T: 0:0
T: 0:1
T: 0:2
T: 0:3
Intro Verse Chorus
Bass
Start: 2 | Loop: 2
T: 0:0:0 P: C2
T: 0:1:2 P: E2
T: 0:1:3 P: F2
T: 0:2:0 P: C2
T: 0:3:0 P: C2
Lead
Start: 3
T: 0:0:0 P: G3 D: 2n
T: 0:2:0 P: G3 D: 8n
T: 0:2:2 P: A3 D: 8n
T: 0:3:0 P: B3 D: 4n

Future Plans

- Write more songs!
- Turn this into an add-on
- Continue to add features
- Collaboratively write songs using GitHub!

Thank You!

jamescdavis.github.io/ember-as-song

twitter: @jamscdavis everywhere else: @jamescdavis