

Advanced Coding Tools and Methodologies

Prof. **Francesco Bruschi**

Prof. **Vincenzo Rana**

**Music and Acoustic
Engineering M.Sc.**

Student

Marco Muraro



**POLITECNICO
MILANO 1863**



SubJuicy 
Subtractive Synthesizer

SubJuicy

Vue3 Web App

Monophonic synthesizer based
on subtractive synthesis

Libraries employed

- **Vue.js**
- **Tone.js**

Web MIDI API is exploited to manage
MIDI device connection and messages



POLITECNICO
MILANO 1863



Vue.js

Progressive JavaScript Framework for building web user interfaces

Approachable

Builds on top of
standard HTML, CSS
and JavaScript

Performant

Truly, reactive
compiler-optimized
rendering system

Versatile

A rich, incrementally
adaptable ecosystem

Vue CLI Standard tooling baseline for Vue ecosystem



POLITECNICO
MILANO 1863



Tone.js

Web Audio Framework for creating interactive music in the browser

On the **high level**, Tone.js offers

- Common DAW features like global transport as well as interesting prebuilt synths and effects of different nature
- High-performance building blocks to create your own synthesizers, effects and complex control signals



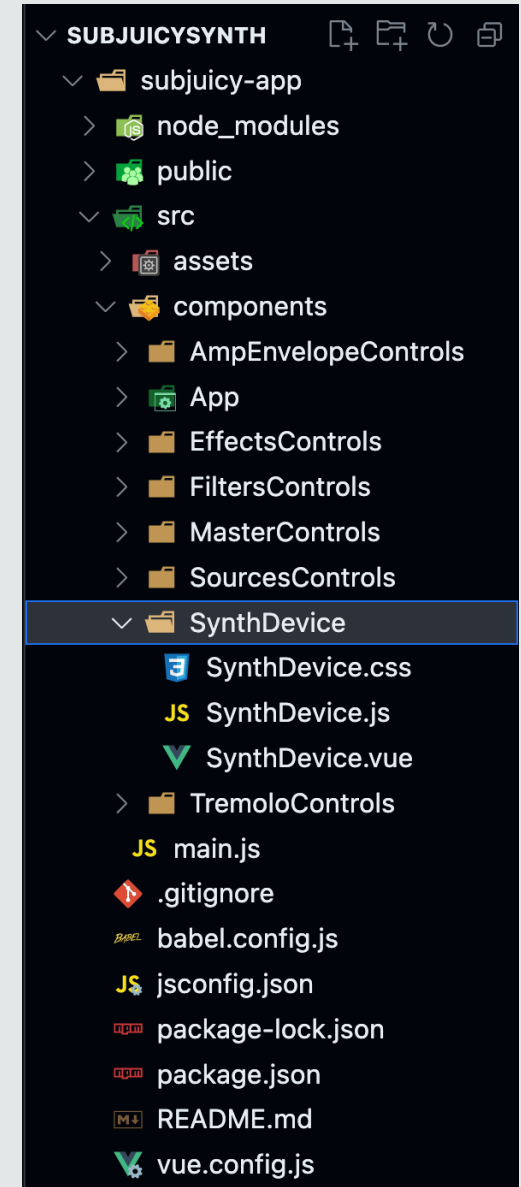
POLITECNICO
MILANO 1863



App Architecture

Functional macro-sections integrated within the GUI of SubJuicy app are implemented as **Vue components**

- **Independent and reusable blocks** that are integrated in the app
- Vue **apps** are commonly **organized into nested components**
- Vue implements its own **component model** that allows us to encapsulate **custom content and logic** in each one of them



Single Component

```
MasterControls.vue X JS MasterControls.js MasterControls.css
subjuicy-app > src > components > MasterControls > MasterControls.vue > {} style scoped
1 <template>
2   <div class="component-wrapper flex-container-col">
3     <div class="title-container">
4       <h2>Master</h2>
5     </div>
6   <div class="flex-container-row controls-wrapper"> ...
48 </div>
49 </div>
50 </template>
51
52 <script src="./MasterControls.js"></script>
53
54 <style src="./MasterControls.css" scoped></style>
55

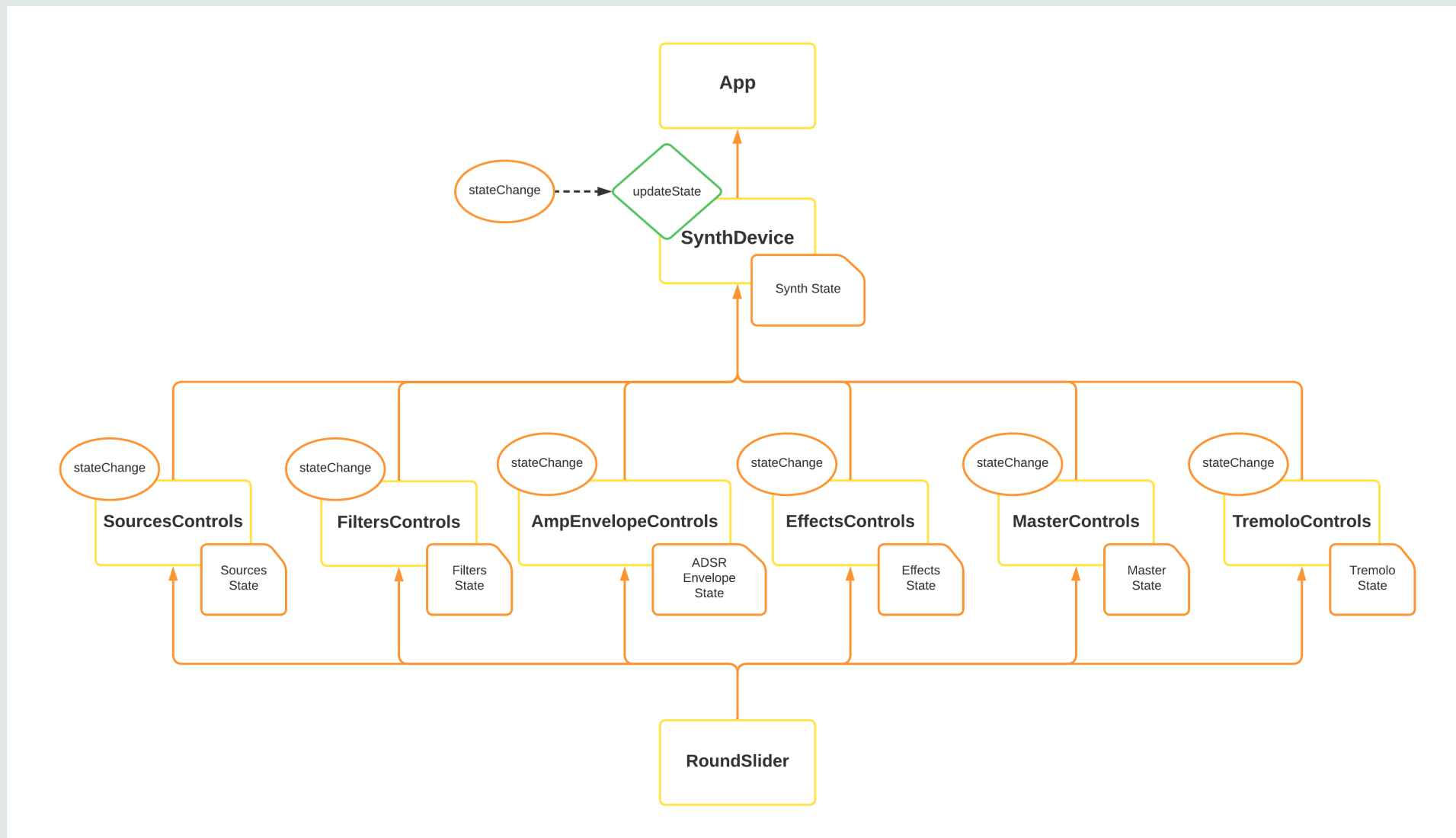
MasterControls.vue JS MasterControls.js X MasterControls.css
subjuicy-app > src > components > MasterControls > JS MasterControls.js > [e] default
1 import RoundSlider from 'vue-three-round-slider'
2
3 export default {
4   name: 'MasterControls',
5   components: {
6     RoundSlider
7   },
8   props: {
9     synthState: Object
10  },
11  data() {
12    return {
13      roundSlider: { ...
23    },
24    masterGain: this.synthState.master.gain,
25    pan: this.synthState.master.pan
26  },
27  },
28  watch: {
29    masterGain() { ...
32  },
33    pan() { ...
36  }
37  }
38 }
39

MasterControls.vue JS MasterControls.js X MasterControls.css
subjuicy-app > src > components > MasterControls > MasterControls.css > ...
1 .component-wrapper {
2   box-sizing: border-box;
3   width: 100%;
4   height: 100%;
5   border-radius: 12px;
6   padding: 15px;
7 }
8
9 .title-container {
10   width: 100%;
11   border-radius: 12px;
12   background-color: #ffc304;
13 }
14
15 .controls-wrapper { ...
18 }
19
20 .knob-wrapper { ...
22 }
23
24 label { ...
28 }
29
30 .unit-of-measure { ...
33 }
34
```





App Architecture

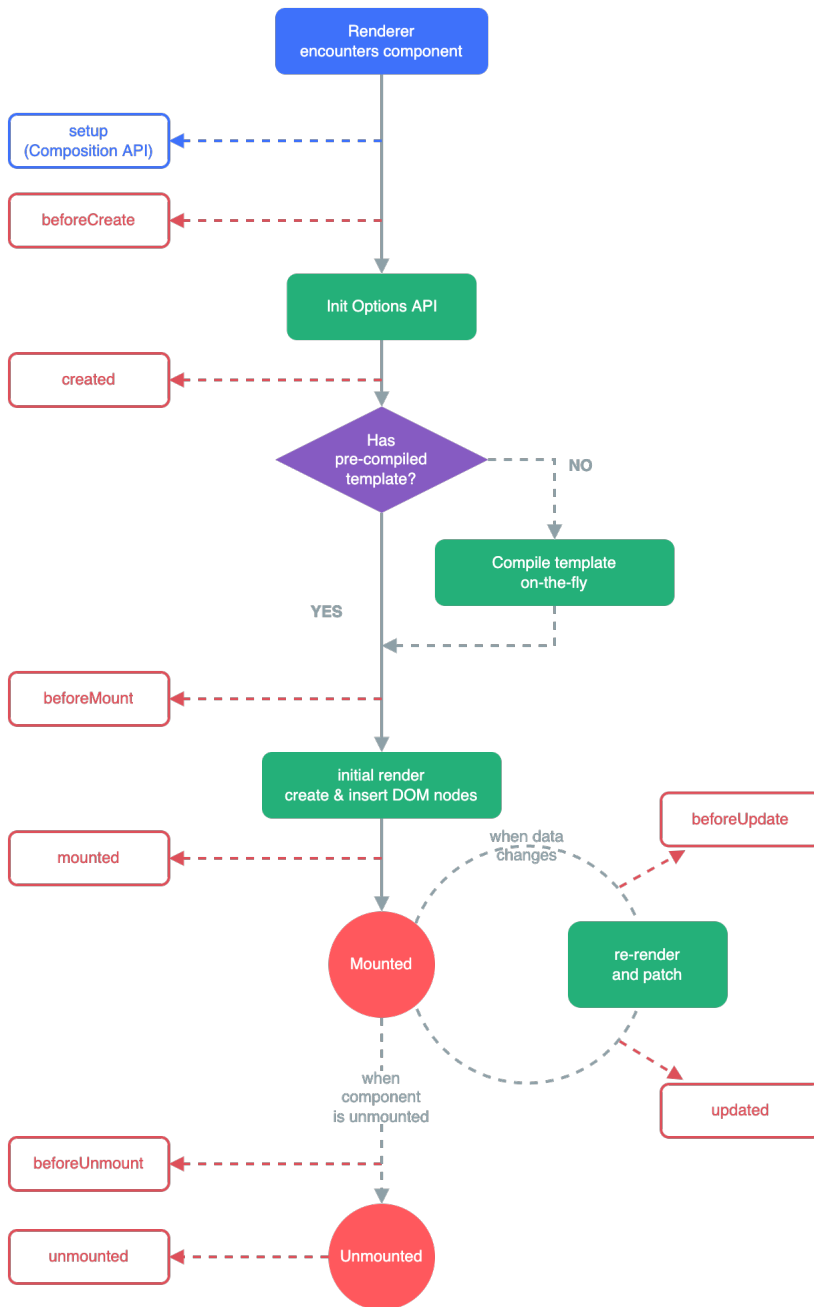


Lifecycle Hooks

Each Vue component goes through a series of initialization steps when it's created and then mounted to the DOM

Along the way, Vue runs functions called **lifecycle hooks**

Developers can add their own code to be executed at specific stages during lifecycle of a certain Vue instance



```
SynthDevice.vue JS SynthDevice.js SynthDevice.css
subjuicy-app > src > components > SynthDevice > JS SynthDevice.js > default > created
404 created() {
405
406     /* ----- CREATE and INITIALIZE AUDIO NODES ----- */
407
408     // Sources
409     this.osc1 = new this.$tone.Oscillator({
410       'type': this.synthState.sources.osc1.type,
411       'volume': this.synthState.sources.osc1.volume
412     });
```



Synth State

Synth state is defined as an object in the **SynthDevice component** within its data function

SynthDevice is the **heart of signal processing** for the whole app

synthState object is exploited to initialize all the audio nodes and is also passed as a property to all the child components for GUI controls initialization

```
synthState: {  
  sources: {  
    osc1: {  
      type: 'sine',  
      volume: 0  
    },  
    osc2: { ...  
    },  
    osc3: { ...  
    },  
    noise: {  
      type: 'white',  
      volume: -50  
    }  
  },  
  amplitudeEnvelope: { ...  
  },  
  master: { ...  
  },  
  lfo: { ...  
  },  
  filters: { ...  
  },  
  effects: { ...  
  }  
}
```

```
this.peakingFilter = new this.$tone.BiquadFilter({  
  'type': 'peaking',  
  'gain': this.synthState.filters.peaking.gain,  
  'frequency': this.synthState.filters.lowPass.frequency,  
  'Q': this.synthState.filters.lowPass.quality,  
  'detune': this.synthState.filters.lowPass.detune  
});
```



State Update

v-model directive automatically updates child component state

Vue watchers are employed to *emit* *stateChange* custom event that is dispatched to the parent SynthDevice whenever the user modifies any synth parameter

```
SourcesControls.vue JS SourcesControls.js SourcesControls.css JS SynthDevice.js
subjuicy-app > src > components > SourcesControls > JS SourcesControls.js > ...
38   watch: {
39     osc1Type() {
40       console.log('Osc1 Type chnaged:', this.osc1Type);
41       this.$emit("stateChange", {name: 'osc1Type', value: this.osc1Type});
42     },
43     osc1Volume() {
44       console.log('Osc1 Volume changed:', this.osc1Volume);
45       this.$emit("stateChange", {name: 'osc1Volume', value: this.osc1Volume});
46     },
```

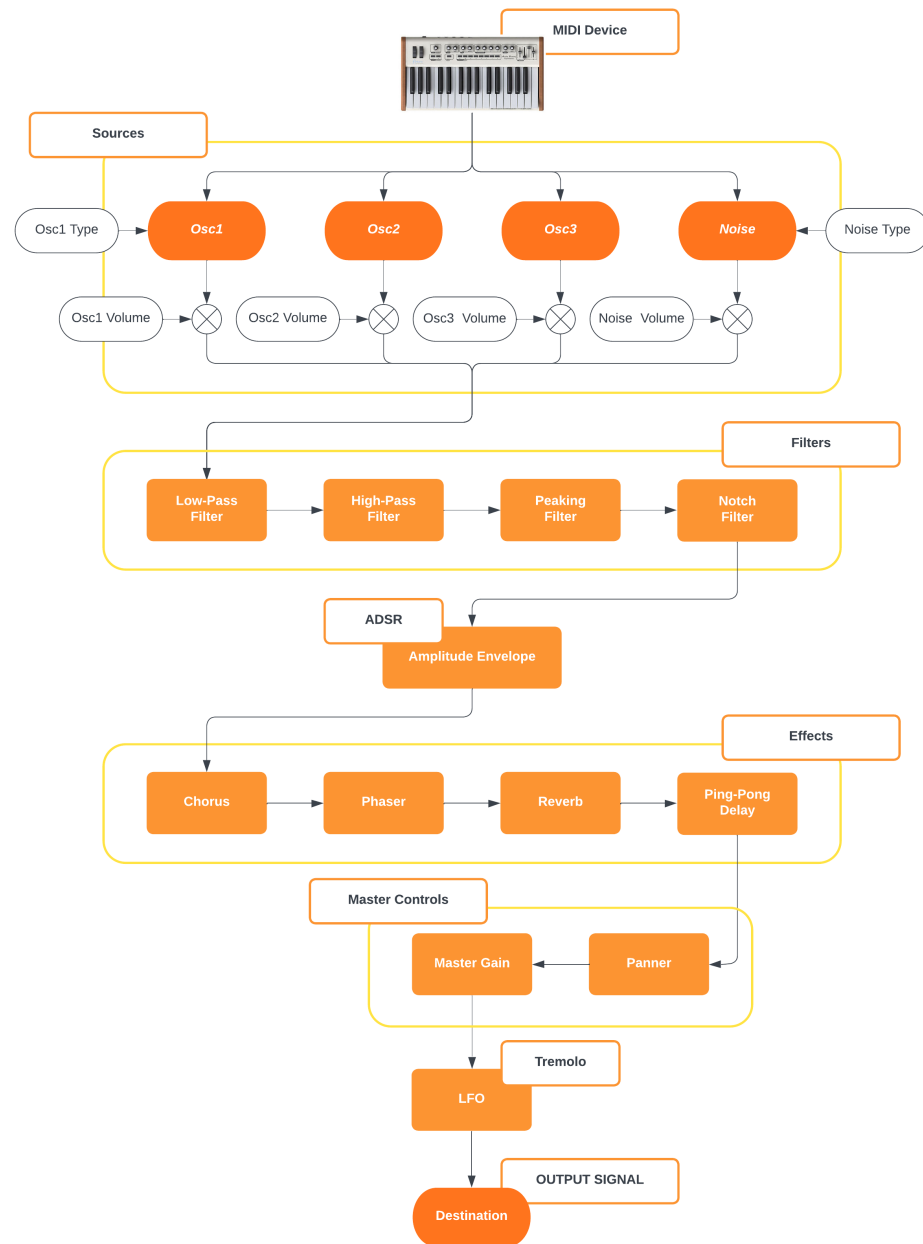
```
SynthDevice.vue JS SynthDevice.js SynthDevice.css JS SourcesControls.js
SynthDevice.vue > {} template > div#synth.flex-container-row > div#synth-panel.flex-container-col
22   <div id="controls-panel" class="flex-container-col">
23     <div class="controls-wrapper flex-container-row">
24       <div id="sources-controls">
25         <SourcesControls
26           :synthState="synthState"
27           @stateChange="updateState"/>
28       </div>
```

```
SynthDevice.vue JS SynthDevice.js
subjuicy-app > src > components > SynthDevice
184   updateState(update) {
185     switch(update.name) {
186       case 'osc1Type':
187         this.synthState.sources.osc1.type = update.value;
188         this.osc1.type = this.synthState.sources.osc1.type;
189         console.log('Osc1 Type updated:', update.value);
190         break;
191       case 'osc1Volume':
192         this.synthState.sources.osc1.volume = update.value;
193         this.osc1.volume.value = this.synthState.sources.osc1.volume;
194         console.log('Osc1 Volume updated:', update.value);
195         break;
```



Signal Flow

Signal processing is implemented through **Tone.js audio nodes** that are connected together to build the overall signal flow architecture



SubJuicy



Filters

Fx

Sources

Osc 1



sine

Osc 2



triangle

Osc 3



square

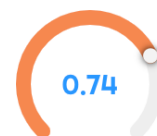
Noise



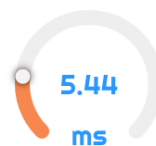
pink

Effects

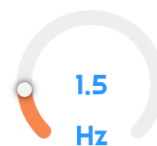
Depth



Delay Time



Frequency



Feedback



Wet



Chorus

Master

Gain

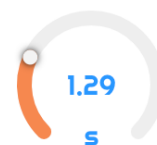


Pan



ADSR Envelope

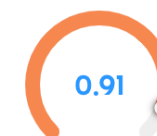
Attack



Decay



Sustain



Release

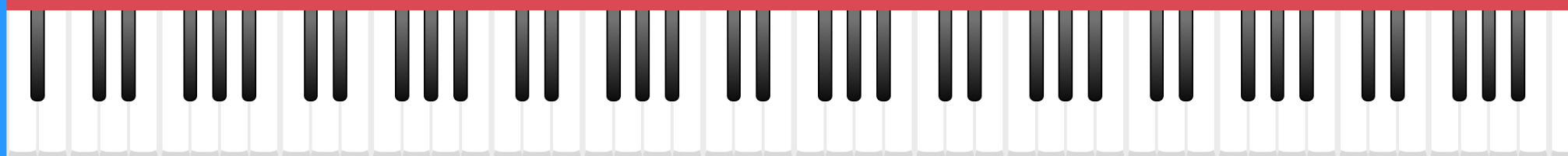
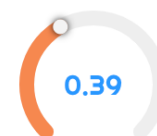


LFO

Frequency



Depth



Let's jump to

SubJuicy



...and make some music

<https://marcomuraro4.github.io/SubJuicy/>

Advanced Coding Tools and Methodologies

Prof. **Francesco Bruschi**

Prof. **Vincenzo Rana**

**Music and Acoustic
Engineering M.Sc.**

Student

Marco Muraro

SubJuicy



Thank You



POLITECNICO
MILANO 1863

