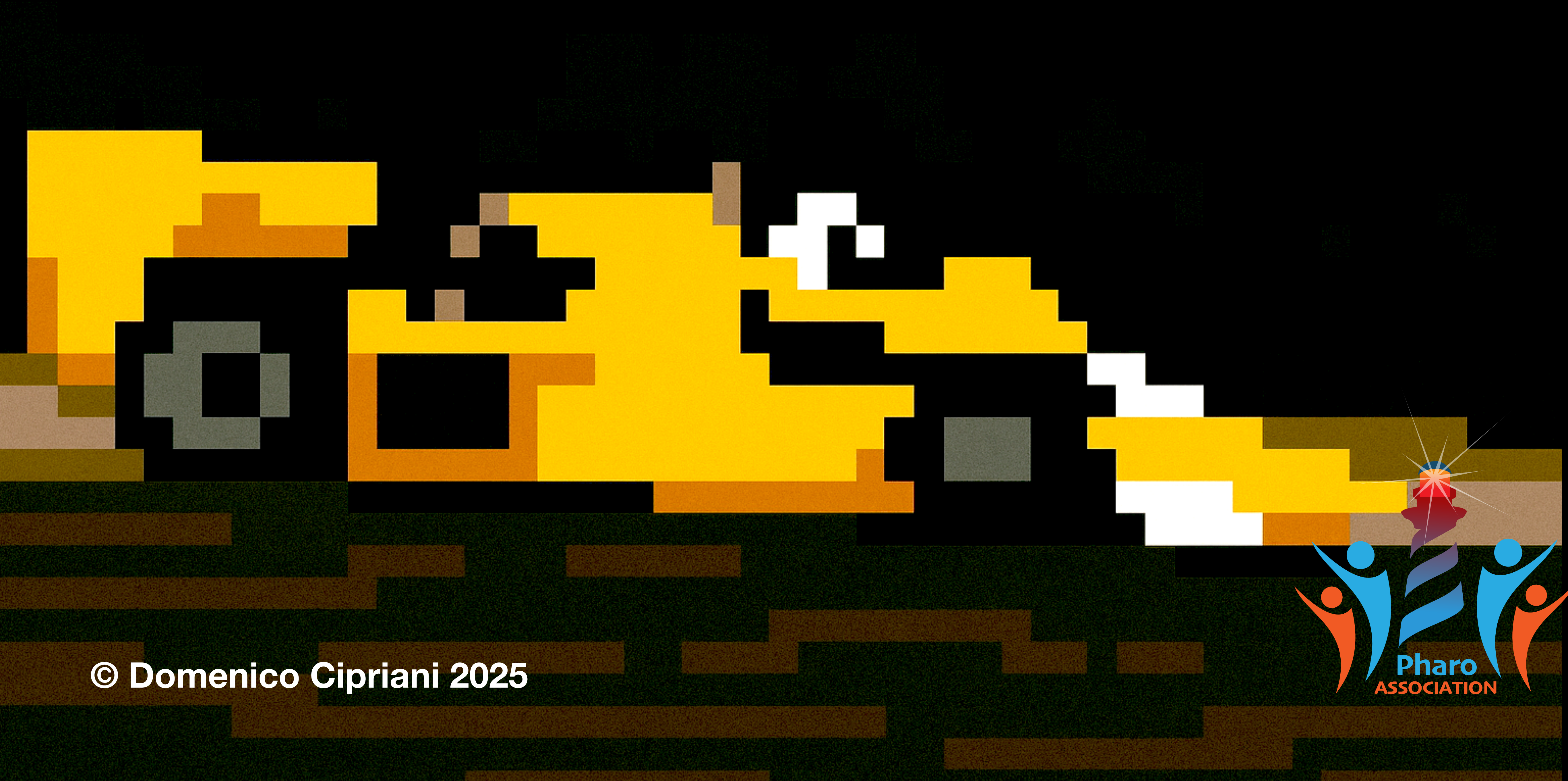
(TURBO)PHAUSTO

News from the pitlane



AGENDA

- What is Phausto?
- What is TurboPhausto?
- Phausto updates
- Future work

What is Phausto?

Phausto is a multi-platform library and API that enables programming Digital Signal Processors (DSPs) and sound generation in Pharo

The audio is generated through FFI calls to a FAUST architecture (the *dynamic engine)* that leverage the power on an embedded FAUST compiler

Phausto has been developed with three main goals:

1. To enrich Pharo applications with sound;
2. To allow sound artists and musician to program synthesisers and effects and compose music with Pharo;
3. To teach DSP programming to beginners and offer a fast prototyping platform for musician and audio developers

What is TurboPhausto?

TurboPhausto is set of synthesisers and effects, *made with Phausto*, especially designed for programming music on-the-fly with Coypu.

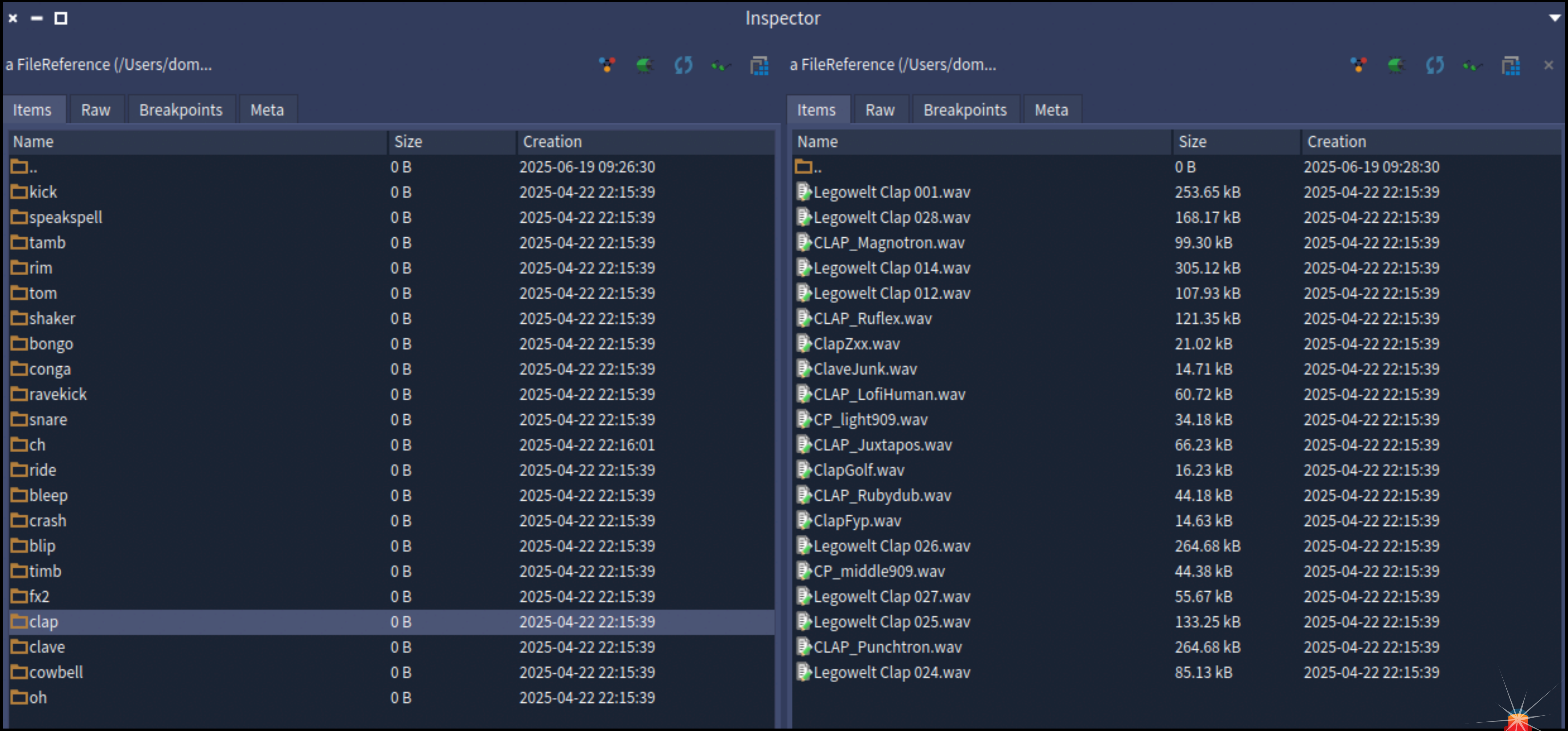
Inspired by the SuperDirt audio engine for SuperCollider, it turns Pharo into a powerful environment for live-coded music and algoraves.

It comes with a folder of 50 MB of high-quality-algorave-ready royalty free audio samples, made by *Lucretio*, *The Analogue Cops* and the legendary dutch electro producer *Legowelt*.

TurboSamples

15 folders of *.wav* samples - each folder has a different number of samples.

```
TurboPhausto listOfSamples.
```



Inspector

a FileReference (/Users/dom...

Items | Raw | Breakpoints | Meta

Name	Size	Creation
..	0 B	2025-06-19 09:26:30
kick	0 B	2025-04-22 22:15:39
speakspell	0 B	2025-04-22 22:15:39
tamb	0 B	2025-04-22 22:15:39
rim	0 B	2025-04-22 22:15:39
tom	0 B	2025-04-22 22:15:39
shaker	0 B	2025-04-22 22:15:39
bongo	0 B	2025-04-22 22:15:39
conga	0 B	2025-04-22 22:15:39
ravekick	0 B	2025-04-22 22:15:39
snare	0 B	2025-04-22 22:15:39
ch	0 B	2025-04-22 22:16:01
ride	0 B	2025-04-22 22:15:39
bleep	0 B	2025-04-22 22:15:39
crash	0 B	2025-04-22 22:15:39
blip	0 B	2025-04-22 22:15:39
timb	0 B	2025-04-22 22:15:39
fx2	0 B	2025-04-22 22:15:39
clap	0 B	2025-04-22 22:15:39
clave	0 B	2025-04-22 22:15:39
cowbell	0 B	2025-04-22 22:15:39
oh	0 B	2025-04-22 22:15:39

a FileReference (/Users/dom...

Items | Raw | Breakpoints | Meta

Name	Size	Creation
..	0 B	2025-06-19 09:28:30
Legowelt Clap 001.wav	253.65 kB	2025-04-22 22:15:39
Legowelt Clap 028.wav	168.17 kB	2025-04-22 22:15:39
CLAP_Magnotron.wav	99.30 kB	2025-04-22 22:15:39
Legowelt Clap 014.wav	305.12 kB	2025-04-22 22:15:39
Legowelt Clap 012.wav	107.93 kB	2025-04-22 22:15:39
CLAP_Ruflex.wav	121.35 kB	2025-04-22 22:15:39
ClapZxx.wav	21.02 kB	2025-04-22 22:15:39
ClaveJunk.wav	14.71 kB	2025-04-22 22:15:39
CLAP_LofiHuman.wav	60.72 kB	2025-04-22 22:15:39
CP_light909.wav	34.18 kB	2025-04-22 22:15:39
CLAP_Juxtapos.wav	66.23 kB	2025-04-22 22:15:39
ClapGolf.wav	16.23 kB	2025-04-22 22:15:39
CLAP_Rubydub.wav	44.18 kB	2025-04-22 22:15:39
ClapFyp.wav	14.63 kB	2025-04-22 22:15:39
Legowelt Clap 026.wav	264.68 kB	2025-04-22 22:15:39
CP_middle909.wav	44.38 kB	2025-04-22 22:15:39
Legowelt Clap 027.wav	55.67 kB	2025-04-22 22:15:39
Legowelt Clap 025.wav	133.25 kB	2025-04-22 22:15:39
CLAP_Punchtron.wav	264.68 kB	2025-04-22 22:15:39
Legowelt Clap 024.wav	85.13 kB	2025-04-22 22:15:39

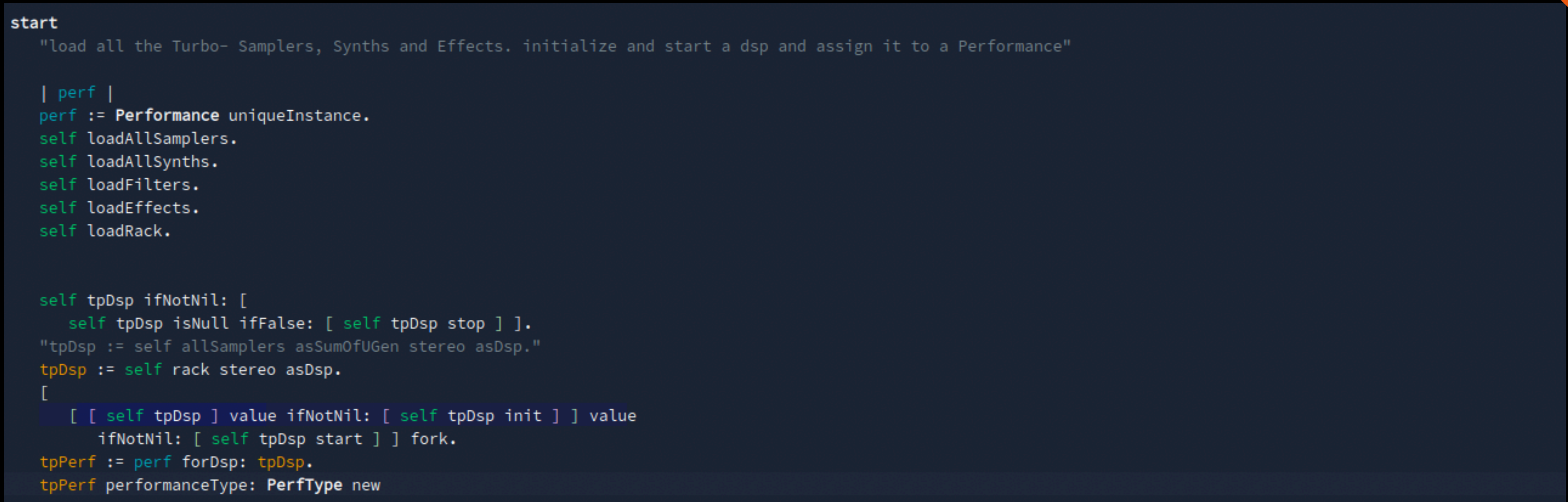
Start your engine

```
TurboPhausto start.
tp := TurboPhausto new.
tp bpm: 167.

'9090' hexBeat to: #kick.
16 cumbiaClave to: #marimba.
16 rumba to: #conga.

tp play.
```

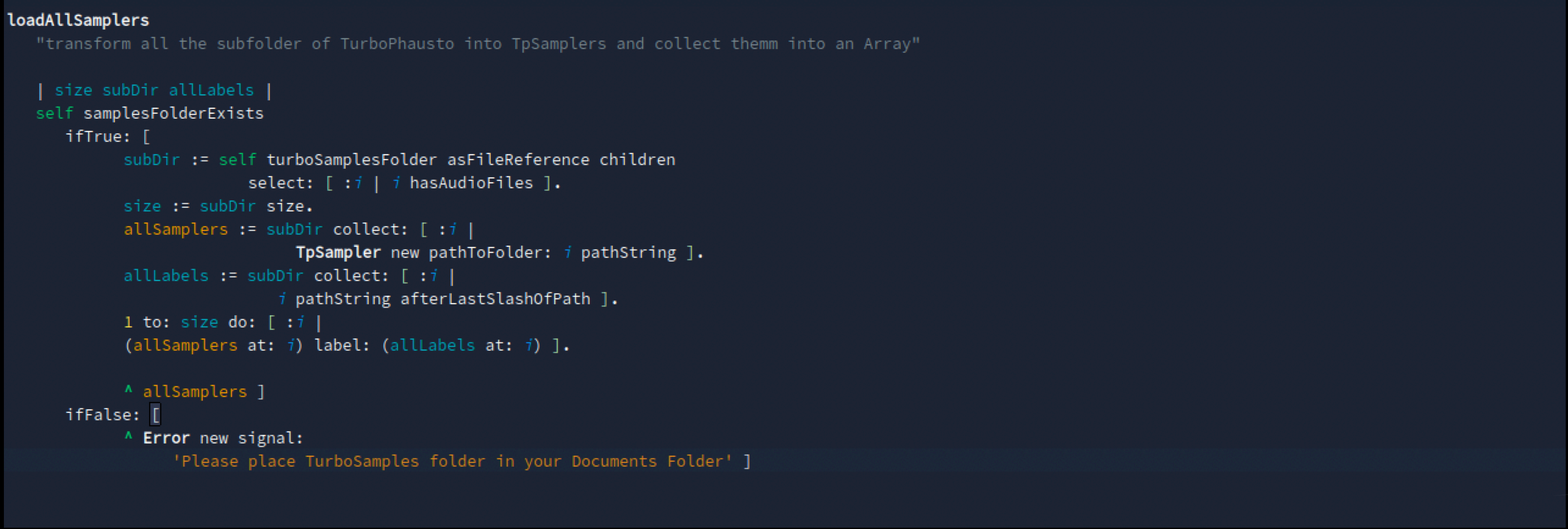
The **`start`** method loads all instruments and effect and create a **DSP** that is assign to a class variable named **`tpDsp`**.

It also initialise a new Performance (also stored in a class variable) and connect it to the DSP.

Peek inside the engine



```
start
   "load all the Turbo- Samplers, Synths and Effects. initialize and start a dsp and assign it to a Performance"

   | perf |
   perf := Performance uniqueInstance.
   self loadAllSamplers.
   self loadAllSynths.
   self loadFilters.
   self loadEffects.
   self loadRack.


   self tpDsp ifNotNil: [
      self tpDsp isNull ifFalse: [ self tpDsp stop ] ].
   "tpDsp := self allSamplers asSumOfUGen stereo asDsp."
   tpDsp := self rack stereo asDsp.
   [
      [ [ self tpDsp ] value ifNotNil: [ self tpDsp init ] ] value
         ifNotNil: [ self tpDsp start ] ] fork.
   tpPerf := perf forDsp: tpDsp.
   tpPerf performanceType: PerfType new
```



```
loadAllSamplers
   "transform all the subfolder of TurboPhausto into TpSamplers and collect themm into an Array"

   | size subDir allLabels |
   self samplesFolderExists
      ifTrue: [
         subDir := self turboSamplesFolder asFileReference children
                     select: [ :i | i hasAudioFiles ].
         size := subDir size.
         allSamplers := subDir collect: [ :i |
                          TpSampler new pathToFolder: i pathString ].
         allLabels := subDir collect: [ :i |
                         i pathString afterLastSlashOfPath ].
         1 to: size do: [ :i |
         (allSamplers at: i) label: (allLabels at: i) ].

         ^ allSamplers ]
      ifFalse: [
         ^ Error new signal:
              'Please place TurboSamples folder in your Documents Folder' ]
```

In pole position

In the last 12 months, we've successfully ported and tested 75% of the Faust standard libraries in Phausto!

- 20 classes from the *Basic library*;
- All the *Oscillators*, *Filters* and *Synths*;
- All the *Reverbs*, ***Delays*** , ***Saturators*** and **Effects**;
- All the ***Modal Percussions***, ***String Instruments*** and ***Wind Instruments***
- All the classes from the *Math library*;
- All the ***Demo Synthesisers*** and ***Effects***

190 tested classes !

More stability

Less crashes, as all **`UnitGenerators`** become **`NullObjects`** after the Faust compilation context is destroyed *(thanks Seba!!!!)*

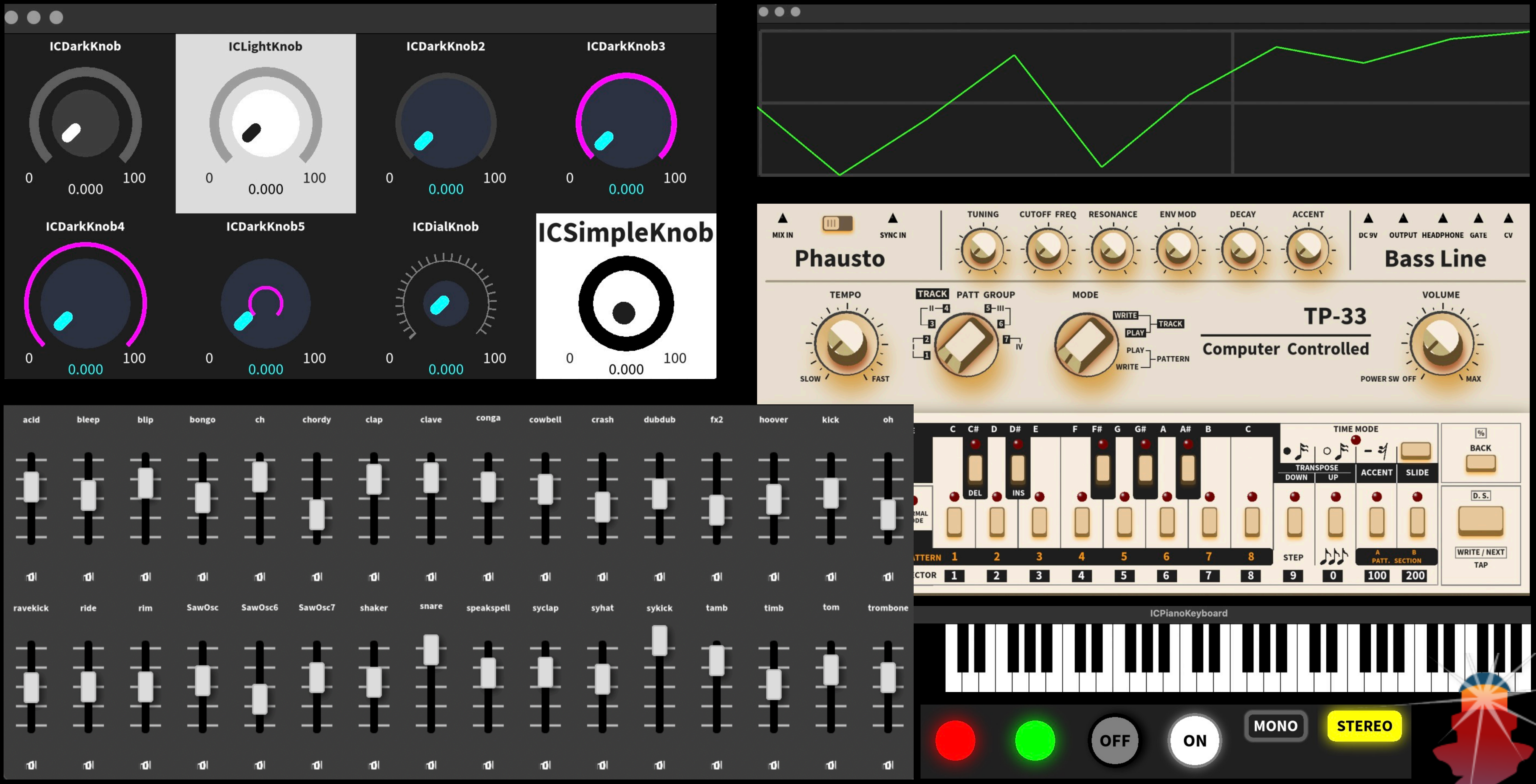
⁉ Phausto crashes are caused by *Faust dynamic-engine* handling of errors, with **C++** try/catch . Achieving full crash-resilience requires refactoring the problematic methods into **C** for safer error handling.

More tools

! Even more tools, beyond Faust standard libraries!

- **`PhCapture`** and **`PhLooper`**: record audio input in real time
- **`PhList`**: allows to create lists of `Number` or `Unit Generators`
- **`PhSelectN`**: select between multiple inputs or elements of a PhList
- **`LFORandomPos`**: a random Low Frequency Oscillator with positive values

Sparkling Bloc UI



© Domenico Cipriani 2025

From Pharo to the DAW

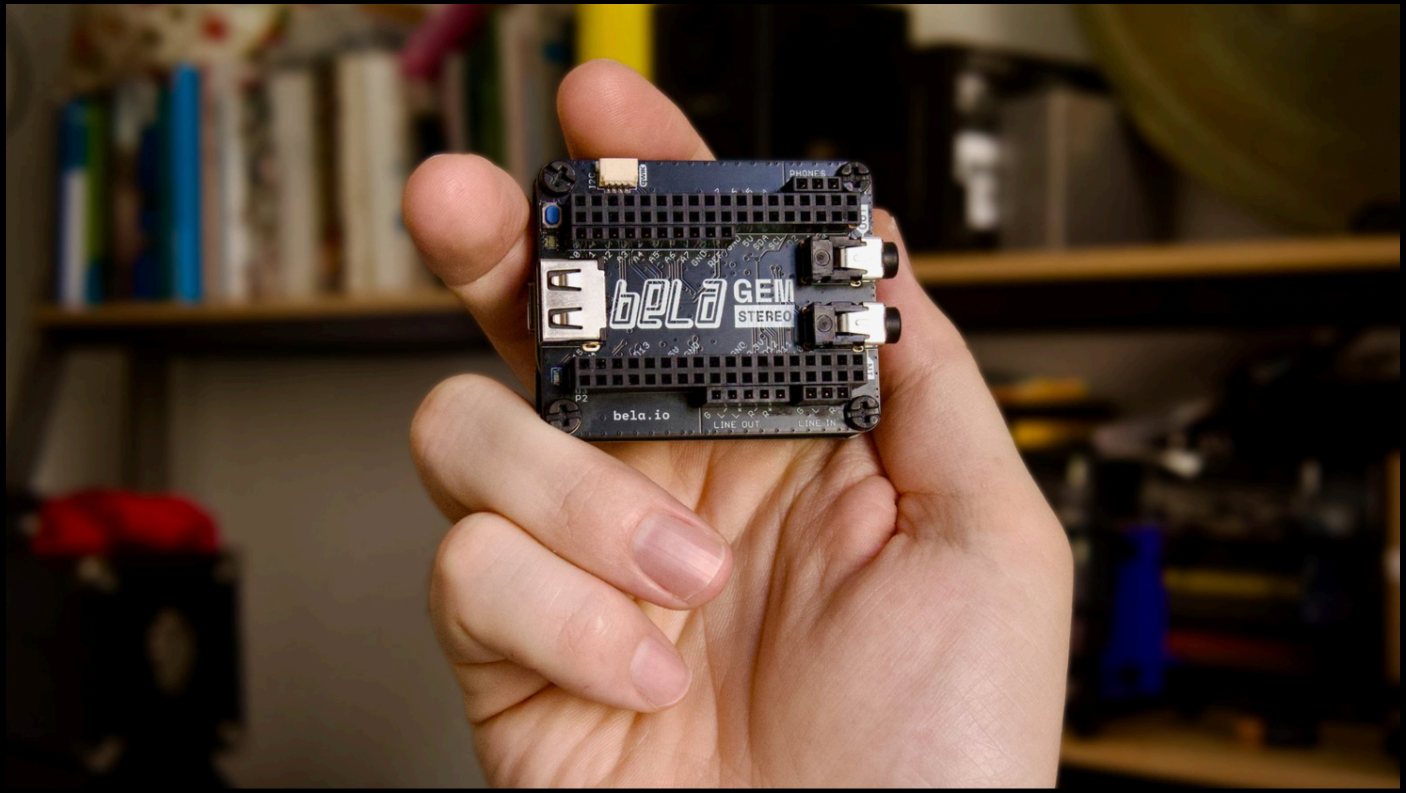
Export *DSP* developed in **Phausto** into a **Cmajor** patch *(thanks Cesare Ferrari)*

The patch can be loaded into the Cmajor wrapper and used in any DAW as a VST3 or AudioUnit plugin.

The patch can be loaded statically or dynamically into any plug-in written in C++

F r o m P h a r o t o B E L A

Export *DSP* developed in **Phausto** into C++ code ready for BELA board

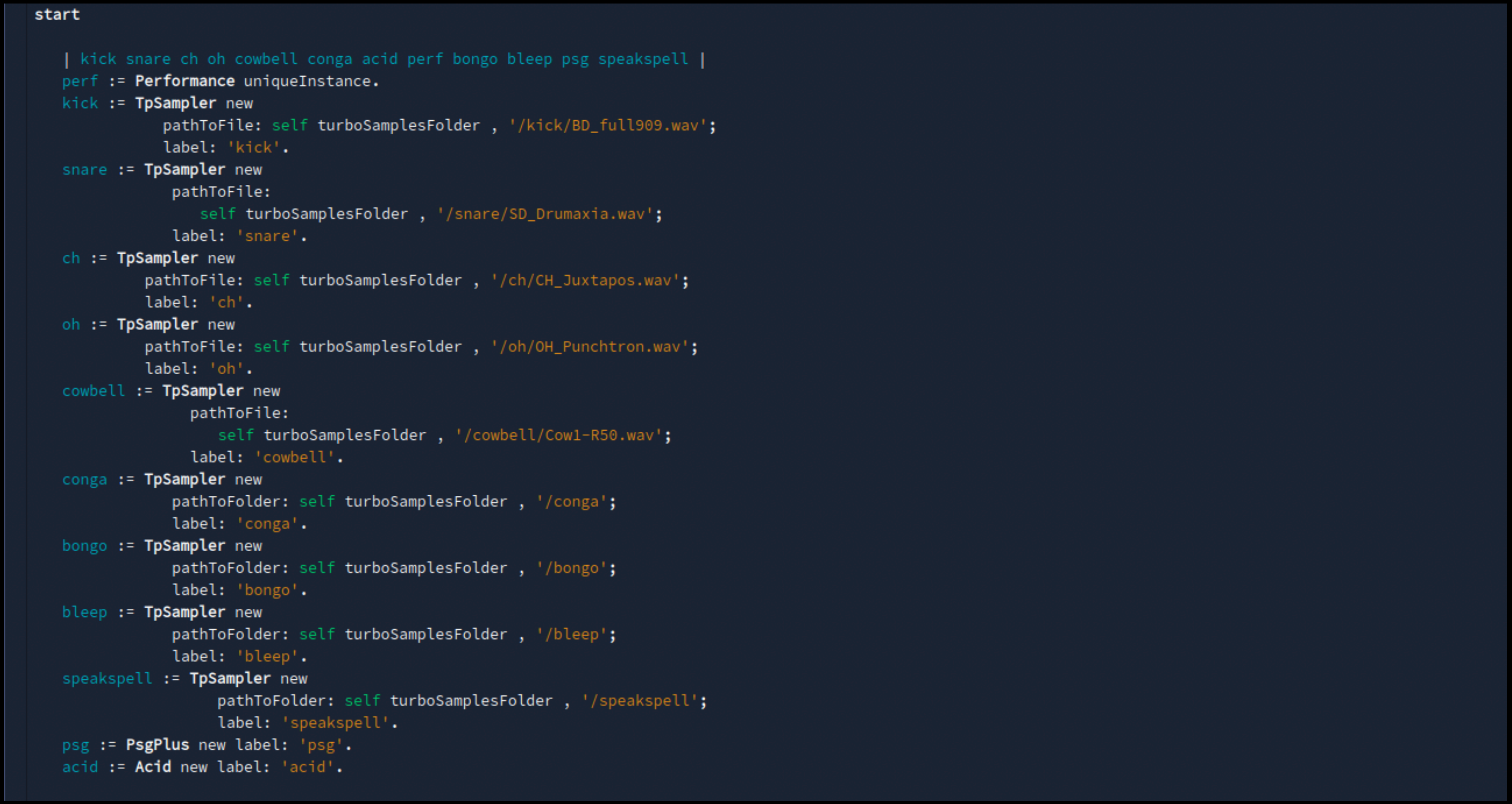


BELA board is an open-source embedded computing platform for creating responsive, real-time interactive systems with audio and sensors.

EcoPhausto?????????

EchoPhausto is a lightweight version of TurboPhausto designed for improved performance on *Windows/Linux* without audio interface.

Less instruments(11) & no effects nor filters on output -> no audio glitches

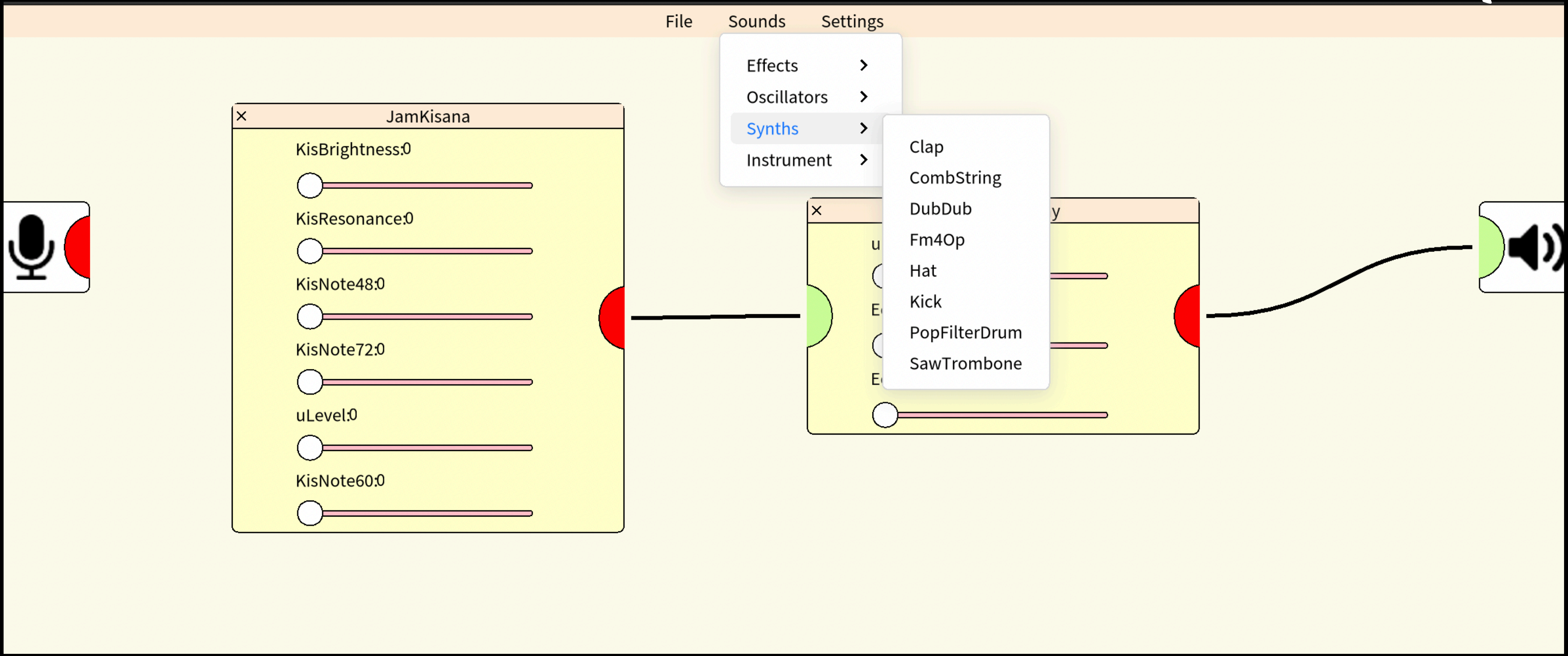


```
start

   | kick snare ch oh cowbell conga acid perf bongo bleep psg speakspell |
   perf := Performance uniqueInstance.
   kick := TpSampler new
              pathToFile: self turboSamplesFolder , '/kick/BD_full909.wav';
              label: 'kick'.
   snare := TpSampler new
               pathToFile:
                  self turboSamplesFolder , '/snare/SD_Drumaxia.wav';
               label: 'snare'.
   ch := TpSampler new
            pathToFile: self turboSamplesFolder , '/ch/CH_Juxtapos.wav';
            label: 'ch'.
   oh := TpSampler new
            pathToFile: self turboSamplesFolder , '/oh/OH_Punchtron.wav';
            label: 'oh'.
   cowbell := TpSampler new
                 pathToFile:
                    self turboSamplesFolder , '/cowbell/Cow1-R50.wav';
                 label: 'cowbell'.
   conga := TpSampler new
               pathToFolder: self turboSamplesFolder , '/conga';
               label: 'conga'.
   bongo := TpSampler new
               pathToFolder: self turboSamplesFolder , '/bongo';
               label: 'bongo'.
   bleep := TpSampler new
               pathToFolder: self turboSamplesFolder , '/bleep';
               label: 'bleep'.
   speakspell := TpSampler new
                    pathToFolder: self turboSamplesFolder , '/speakspell';
                    label: 'speakspell'.
   psg := PsgPlus new label: 'psg'.
   acid := Acid new label: 'acid'.
```

PharoJamSession

JamSession is a new tool modelled after the Faust Playground that enables graphical connections of ready-made instruments and effects.

Made with Bloc and Toplo !

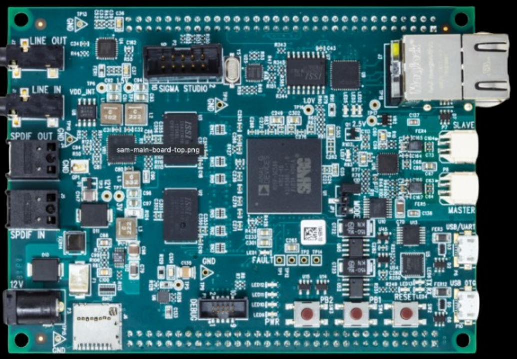


Development started in April during an **Evref** internship by Océane Dubois.

© Domenico Cipriani 2025

The wind tunnel

- 100% coverage of the Faust standard library;
- Graphical representation of the **UGen** connections within a **DSP**;
- Sequencers for the Toolkit;
- More flexible exporters for Cmajor



- Additional exporters *(to RNBO, to SuperCollider, to SHARC AM, …)*;
- Flexible API for UI widget creation and integration with **Bloc/Toplo.**

! Other improvements require rewriting the C++ code of the dynamic-engine !

- No more crashes;
- Full MIDI and OSC integration;

Grazie a voi
e a tutti i
meccanici
di Phausto