



HAL
open science

Phausto: fast and accessible DSP programming in Pharo

Domenico Cipriani

► **To cite this version:**

Domenico Cipriani. Phausto: fast and accessible DSP programming in Pharo. Audio Developer Conference 2024, Nov 2024, Bristol, United Kingdom. hal-04813572

HAL Id: hal-04813572

<https://hal.science/hal-04813572v1>

Submitted on 2 Dec 2024

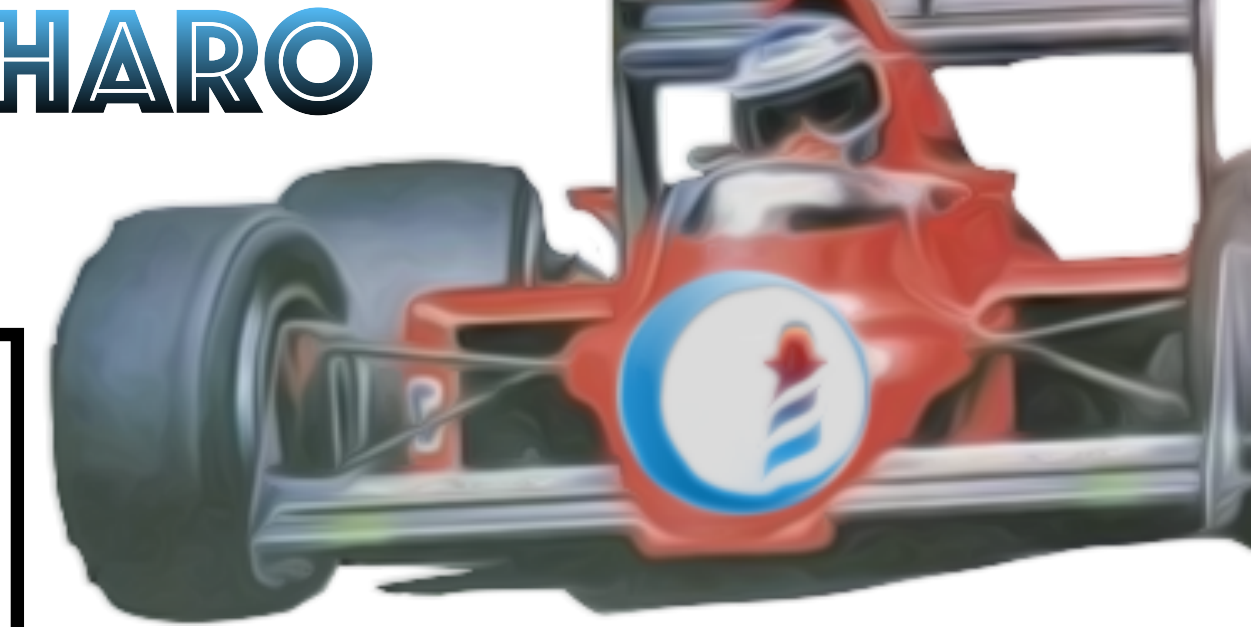
HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

Public Domain

PHAUSTO: FAST AND ACCESSIBLE DSP PROGRAMMING IN PHARO

Domenico Cipriani
Pharo Association



Phausto is a library and API for **Pharo** that enables sound generation and audio *Digital Signal Processing* programming in **Pharo**. **Phausto** leverages a dynamic library accessed via Foreign Function Interface (FFI) calls within Pharo. This library processes synthesizers and effects defined in Phausto with the help of an embedded **FAUST** compiler, which handles real-time audio computation.

Pharo a pure object-oriented programming language and a powerful environment, focused on simplicity and immediate feedback. It is distributed with a non-viral open-source MIT license.

Pharo is a state-of-the-art, modern, cross-platform implementation of the classic **Smalltalk-80** programming language and run time system

- *Easy to learn!*
- *Everything is an object!*
- *Only 6 reserved keywords!*
- *Platform-independent UI!*
- *Syntax fit on a postcard!*
- *Run-time reflection!*
- *Dynamic inheritance!*
- *Integrated Git support!*



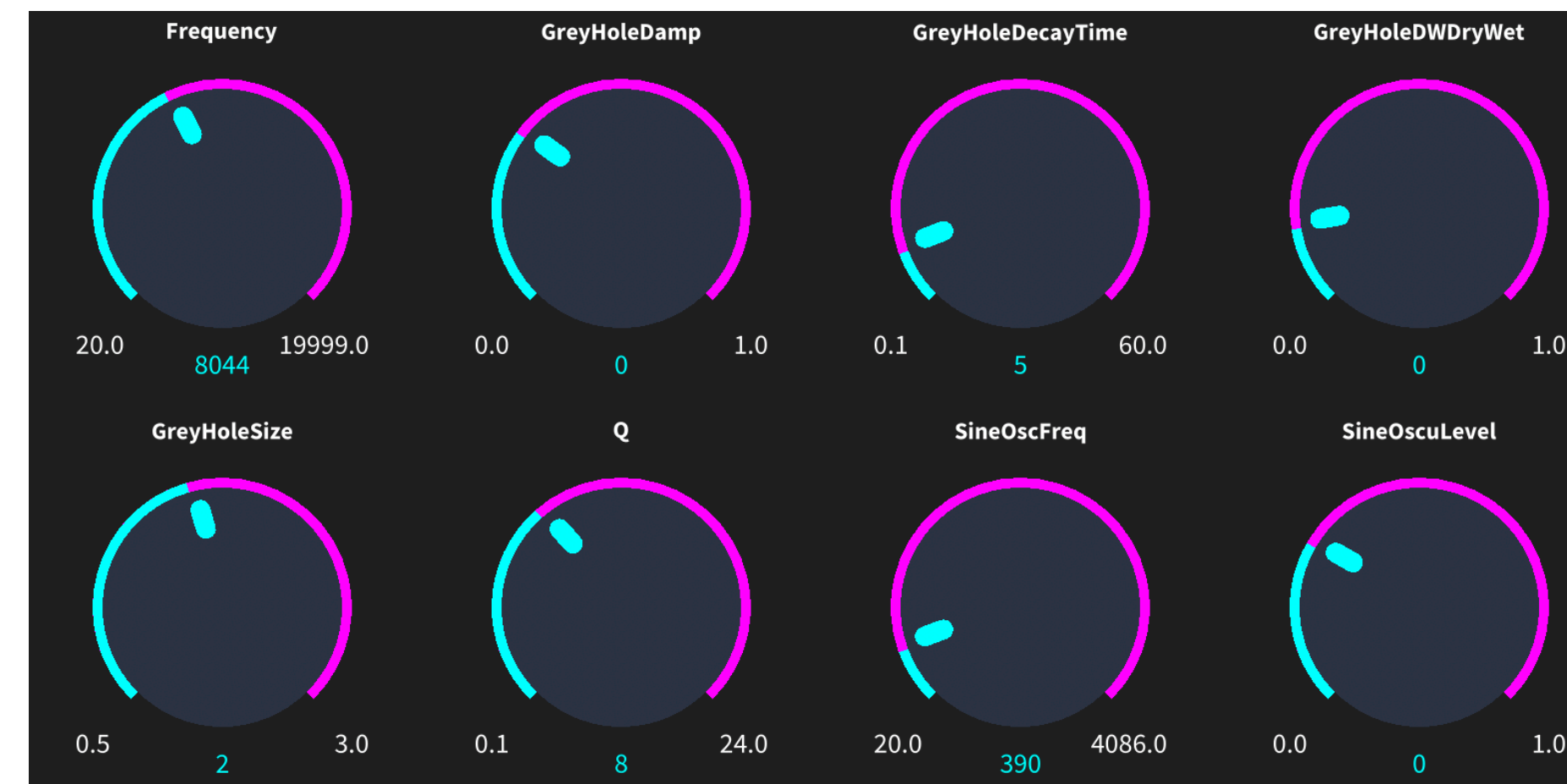
Access a suite of cutting edge functions and generators from **Faust** standard library, including:

- Oscillators and Filters;
- Envelopes;
- Compressors,
- Delays and Reverbs;
- Physical Models;
- Saturators;
- Noises;
- Flangers and phasers;
- And much more

Our grandstand audience

- Artists with little or no computer literacy who want to design and develop synthesizers and effects for their music creation fast and easily.
- Students and beginners who want develop their own audio plug-ins, by exporting DSPs developed in Pharo to a **CMajor** patch thanks to the **Faust-2CMajor** export.
- Pharo programmers who wants to include sounds or sonic interaction in their Pharo applications, including generators coming from **Faust** extensive physical modelling synthesis library.

Easy to display UI made with **Bloc**, a low-level UI infrastructure & framework for **Pharo**.



```
1 dsp := (SineOsc new => GreyHoleDW new => ResonBp new) asDsp.  
2 dsp init.  
3 dsp start.  
4 dsp displayUI |
```

```
1 kick := Kick new => SatRev new.  
2 marimba := Marimba new => DelayMonoFB new.  
3 synth := PulseOsc new => ADSREnv new => MoogVcf new.  
4 violin := ViolinModel new.  
5 psg := PsgPlus new.  
6 dsp := (kick + marimba + synth + violin + psg ) stereo asDsp .  
7 dsp init.  
8 dsp start.  
9 dsp createCmajorPatchNamed: 'PosterSynth'
```

- Develop custom audio plugins through the **Cmajor** exporter. Once the Cmajor files are generated, use them in a DAW with the **Cmajor** VST plugin wrapper,.
- Export DSPa made in **Phausto** into C++ code suitable to build and run on **Bela**,