



Minisoap

A modular audio processor



3/4 people

Project Description

An audio stream processor that can create and transform audio streams, featuring basic synthesis capabilities, soundcard input/output, some MIDI support and a GUI. It is designed modularly, allowing the user to specify its own processing pipeline.

Skills

Languages



Real-time



Audio



(*)[The skill scale is from 0 (Fundamental Awareness) to 6 (Expert).]

Level 1 You may now pursue to the level 1 of the project.

Stream processors

Propose a notion of (audio) stream generator whose parameters (e.g. frequency) can be modified in realtime.

*

Basic generators

Implement a silence generator, and a sine generator parameterized by its frequency and amplitude.

**

Input/Output

Implement a source that reads from the soundcard, and make it possible to output a stream processor on the soundcard.

**

Vumeter

A graphical vumeter should allow the visualization of the signal.



Minisoap

A modular audio processor



3/4 people

Project Description

An audio stream processor that can create and transform audio streams, featuring basic synthesis capabilities, soundcard input/output, some MIDI support and a GUI. It is designed modularly, allowing the user to specify its own processing pipeline.

Skills

Languages



Real-time



Audio



(*)[The skill scale is from 0 (Fundamental Awareness) to 6 (Expert).]

Level 2

Level 1 must be unlocked to read this section

- * Developer Documentation Required for lvl 2 validation
Document your project (not necessarily only in the source code) so that a newcoming developer could understand and contribute to the code.
- * Release Required for lvl 2 validation
Produce a release as a source archive or git tag. The release files should have up-to-date README and INSTALL files and more generally allow anyone to deploy the application.
- ** MIDI input
It must be possible to play MIDI files, using synthesizers that can be set to arbitrary circuits taking note parameters as inputs.
- * Graphical circuit creation
The GUI should offer a user-friendly way to create and display circuits.
- *** Extensible GUI interface
The kernel should export a list of available sources with their parameters, so that e.g. the GUI can make them available to the user for creating and configuring circuits.
- ** Property-based testing
Unit tests should check that amplitude/frequency characteristics of simple signals are as expected. This approach should be applied to basic signal generators, but also to transformations (e.g. mixer, envelope).