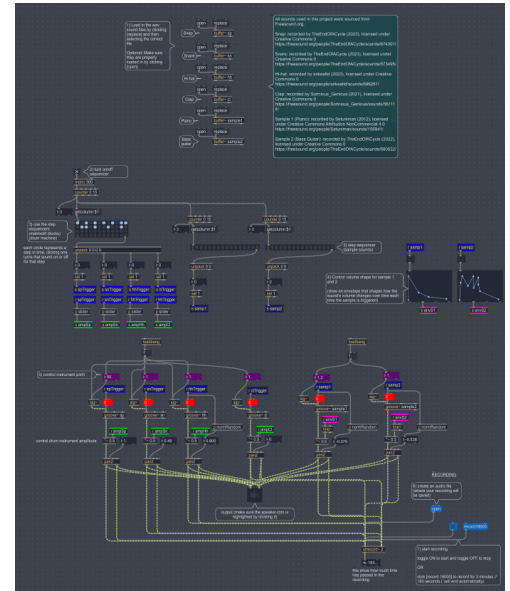


Assignment 4 - Sample-based drum machine and sequencer

Introduction

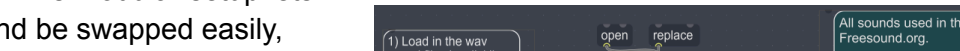
This patch is a custom-built step sequencer instrument created in Max/MSP. It allows users to load, trigger, and manipulate drum and instrument samples to build layered rhythmic compositions. The patch combines multiple sequencing systems, envelope shaping tools, and stereo controls to offer a flexible and performative environment for sound experimentation.

The instrument is structured in modular sections that handle sound loading, sequencing, playback, amplitude shaping, panning, and recording. The goal was to design a patch that feels intuitive and visually clear, encouraging playful exploration while still offering control over key musical parameters.



Part 1: Loading and Organizing Sounds

The top left section of the patch handles sound loading. Each sound slot, Snap, Snare, Hi-hat, Clap, Piano, and Bass Guitar, uses a combination of open, replace, and buffer~ objects. Clicking “replace” allows the user to load their own WAV files, which are stored in memory for playback. I added the open object so the user can check if their sound has been properly loaded in. This modular setup lets each sound be swapped easily, encouraging experimentation.

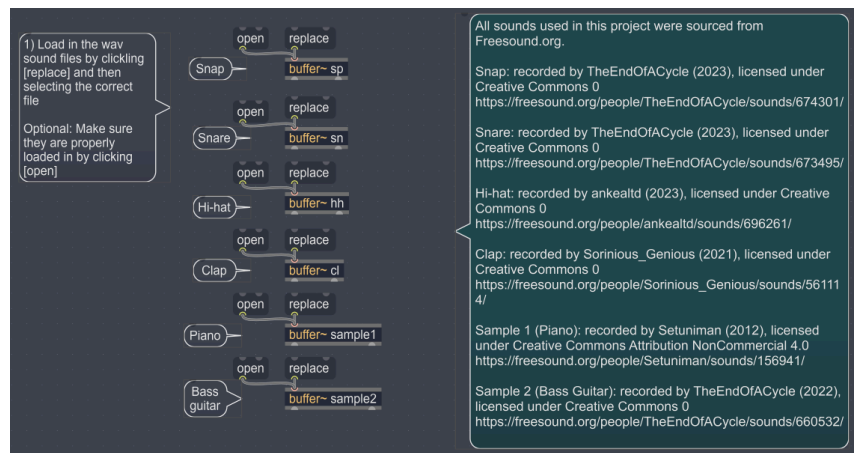


1) Load in the wav sound files by clicking [replace] and then

all sounds used in this project were sourced
Freesound.org.

Snap: recorded by TheEndOfACycle (2023)

All sounds were sourced from [Freesound.org](https://www.freesound.org) and properly credited in the patch. Only one sound (the piano sample) is licensed under Creative Commons Attribution NonCommercial 4.0, while the others are Creative Commons 0 (public domain).

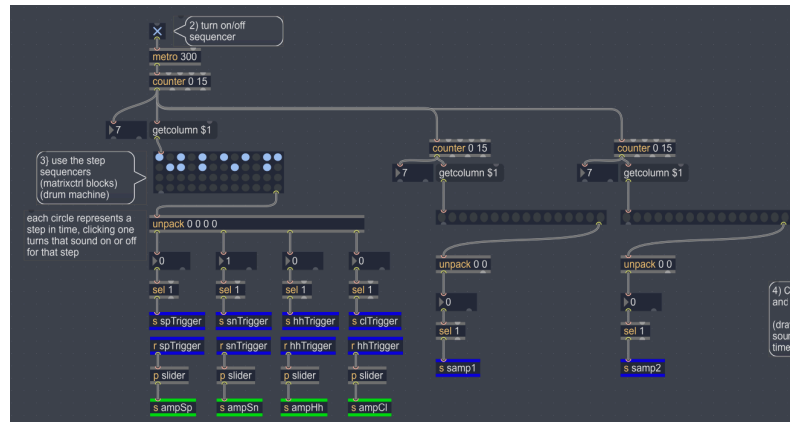


Part 2: Sequencer System

The rhythmic core of the instrument uses `matrixctrl` objects, which serve as 16-step sequencers. Each row corresponds to a different sound, and each column represents a moment in time (a beat step). Clicking on a cell activates or deactivates that sound for that step, visually turning the cell on or off.

A metro 300 object drives the sequence timing by sending bangs at regular intervals (every 300 ms). This clock signal triggers a counter 0 15, cycling through 16 steps in a loop. The getcolumn \$1 and unpack objects read the active steps from each matrix, sending trigger messages (e.g. s spTrigger, s snTrigger) to start playback of the corresponding sound.

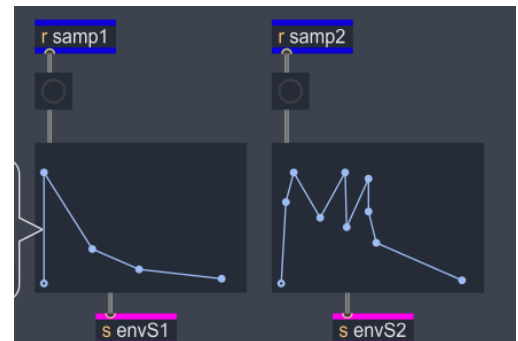
This structure effectively turns the patch into a small drum machine that can be programmed in real time by toggling steps on and off.



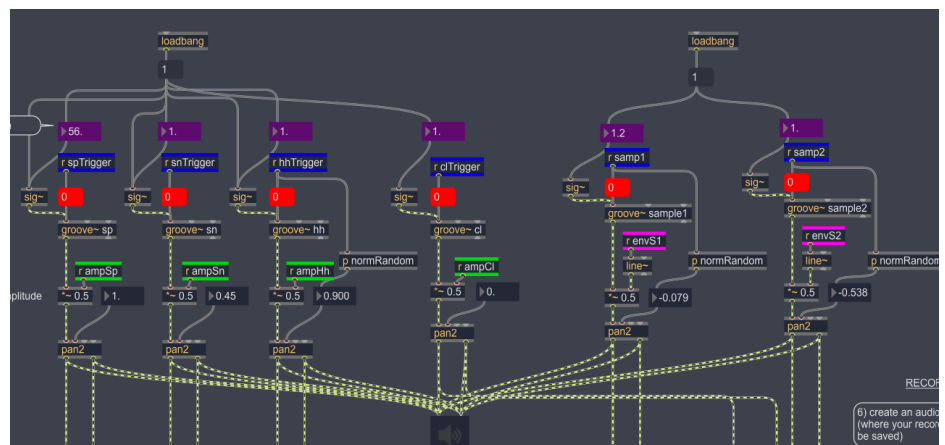
Part 3: Envelope and Playback Control

Two of the sample slots (Piano and Bass Guitar) include breakpoint function envelopes connected to `line~` objects. These envelopes control amplitude over time, allowing each triggered note to have a defined attack, sustain, and decay shape. The user can draw their own volume curves directly in the function window, giving expressive control over how each sample behaves dynamically.

The `r envS1` and `r envS2` objects receive envelope data for the two samples, shaping how they fade in or out when triggered. This helps the instrument feel more alive and less mechanical.



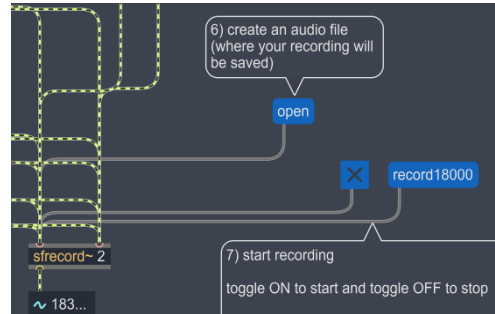
The pan2 objects handle stereo placement for each sound, slightly spreading instruments in the stereo field for a wider mix.



Part 4: Recording System

At the bottom right of the patch, the `srecord~ 2` object allows stereo recording of the final output. The “2” specifies that it records two audio channels (left and right).

Users can click open to create and name a file where the recording will be saved. Then, by toggling the recording control ON, the session begins capturing sound. There’s also an automated `record18000` button that records for three minutes (180 seconds) and stops automatically. A timer below the recorder shows how long the current recording has been running.



Sound File Sources

All sounds used in this project were sourced from [Freesound.org](https://freesound.org).

- Snap: recorded by TheEndOfACycle (2023), licensed under Creative Commons 0 <https://freesound.org/people/TheEndOfACycle/sounds/674301/>
- Snare: recorded by TheEndOfACycle (2023), licensed under Creative Commons 0 <https://freesound.org/people/TheEndOfACycle/sounds/673495/>
- Hi-hat: recorded by ankealtd (2023), licensed under Creative Commons 0 <https://freesound.org/people/ankealtd/sounds/696261/>
- Clap: recorded by Sorinious_Genius (2021), licensed under Creative Commons 0 https://freesound.org/people/Sorinious_Genius/sounds/561114/
- Sample 1 (Piano): recorded by Setuniman (2012), licensed under Creative Commons Attribution NonCommercial 4.0 <https://freesound.org/people/Setuniman/sounds/156941/>
- Sample 2 (Bass Guitar): recorded by TheEndOfACycle (2022), licensed under Creative Commons 0 <https://freesound.org/people/TheEndOfACycle/sounds/660532/>

Affordances and Performance

The step sequencer invites rhythmic play by letting users toggle beats in real time, visually responding through light-up cells. The function envelopes afford dynamic expression, shaping each note’s amplitude by simply drawing curves. The sliders for amplitude, pitch, and panning provide tactile control, while the modular design encourages sound exploration: users can swap samples, change timing, and layer multiple rhythms easily.

Together, these affordances make the patch not just a fixed tool but a performative instrument, something that rewards experimentation and improvisation.

Conclusion

This Max/MSP instrument combines sequencing, sound design, and live control into a cohesive system for rhythmic experimentation. It merges technical structure with creative flexibility, offering users a hands-on way to build, modify, and record their own rhythmic performances directly within Max.