Digital Synthesis and Live Coding

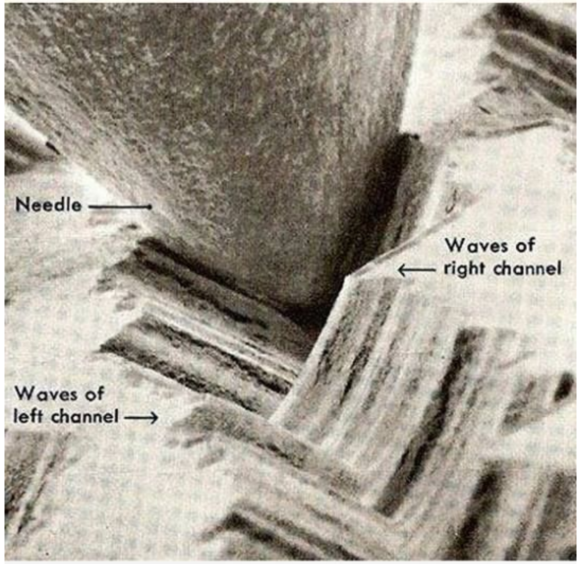
MUS 2830, Interactive Music, Week 4
Qichao Lan
March 1, 2022

Sound is a vibration

- How much?
  - Amplitude/loudness
- How fast?
  - Frequency/pitch
- Timbre

Tuning Fork In Water - Ultra Slow Motion Walking Water Effect - 30,000 FPS

https://www.youtube.com/watch?v=iRYWmo3Tuq4&ab_channel=WarpedPerception


Needle
Waves of right channel
Waves of left channel

Vinyl

https://youtu.be/kUlu-XjCgtk?t=50

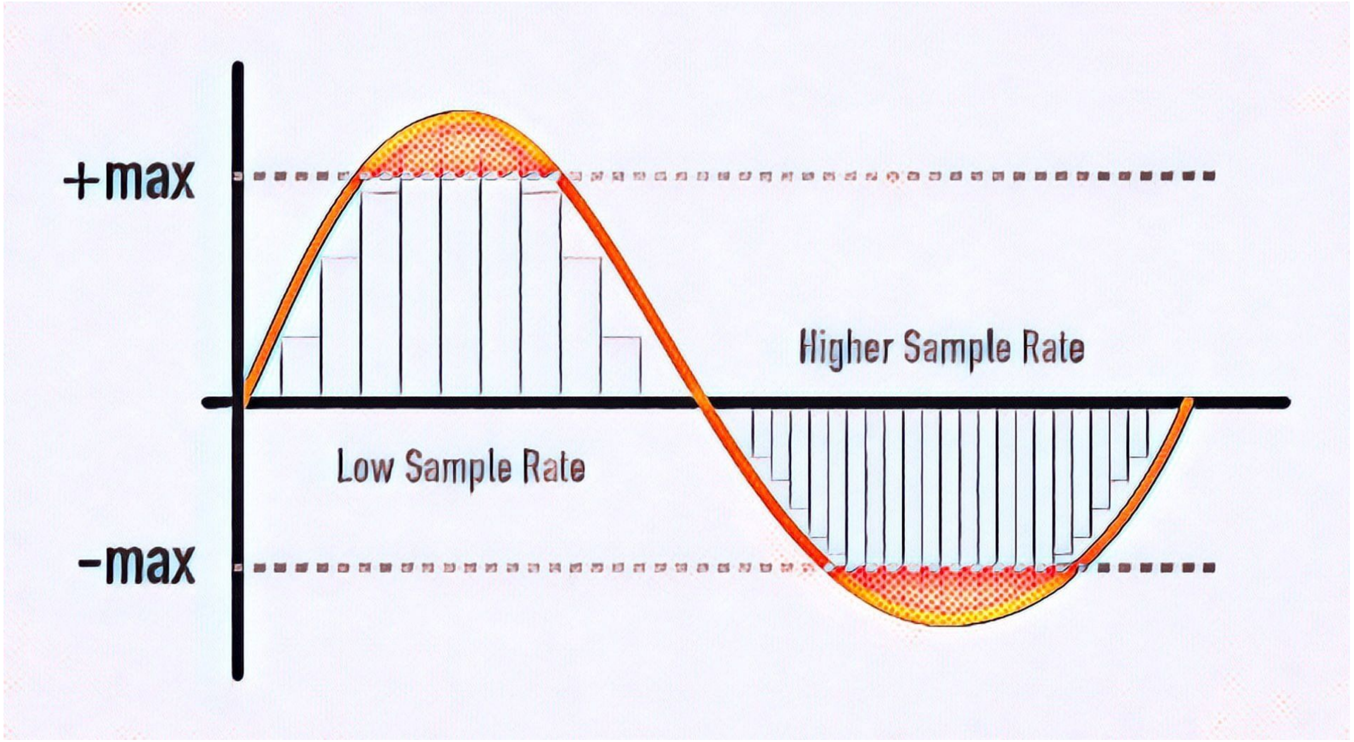
Classroom interaction

- Time
- Space

Digital Sampling

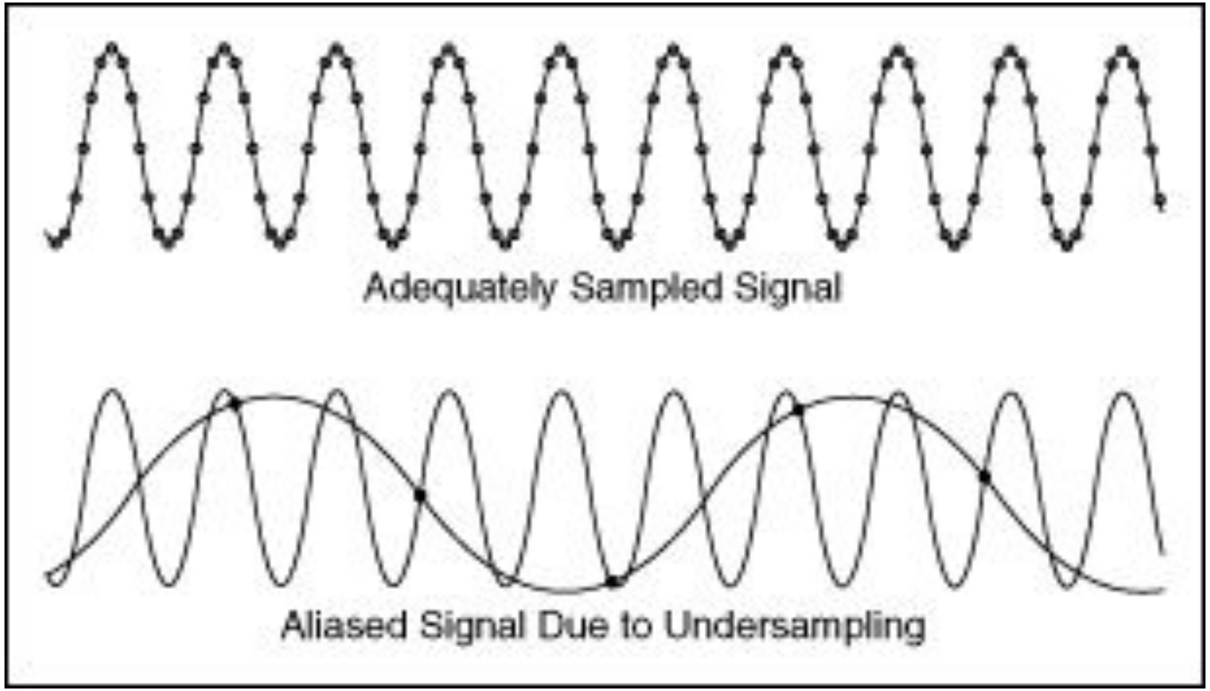
- Sample rate (44.1kHz, 48kHz, 96kHz, 200Hz, 100Hz)
- Bit depth (8-bit, 16-bit, 32-bit)
- Real-time v.s. non-real-time

Sample rate



https://www.headphonesty.com/2019/07/sample-rate-bit-depth-bit-rate/

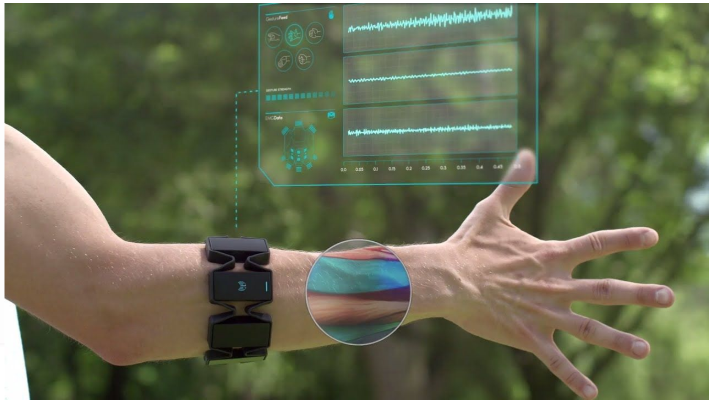
Aliasing


Adequately Sampled Signal
Aliased Signal Due to Undersampling

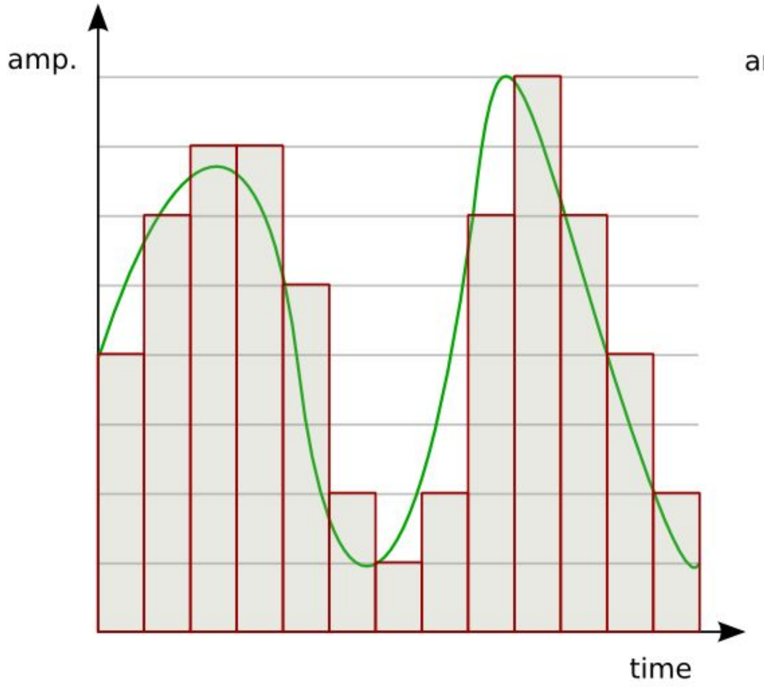
https://zone.ni.com/reference/en-XX/help/375857B-01/mstudioanalysis/aliasing/

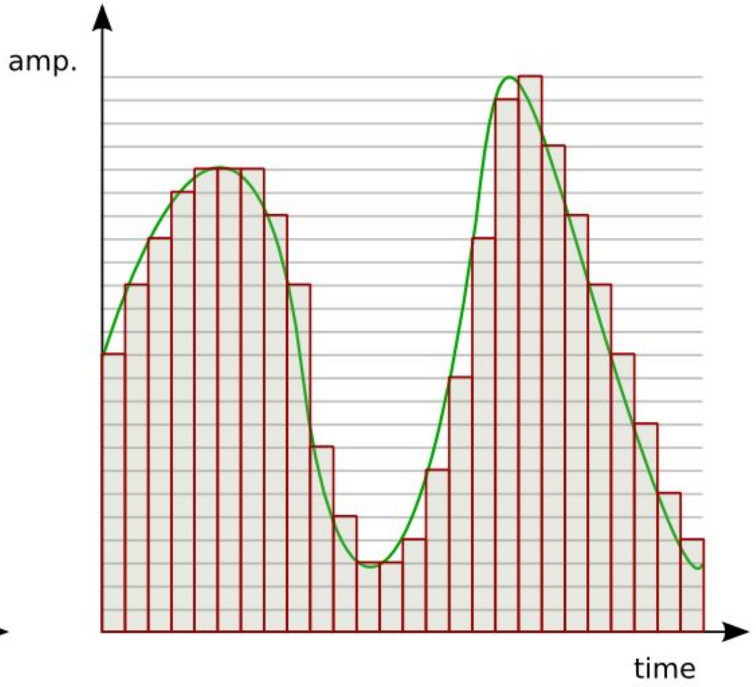
Sample rate

- Audio: 44.1kHz, 48kHz, 96kHz
- ADC: Analog-to-digital converter
  https://youtu.be/dYu55YZJH_s?t=127



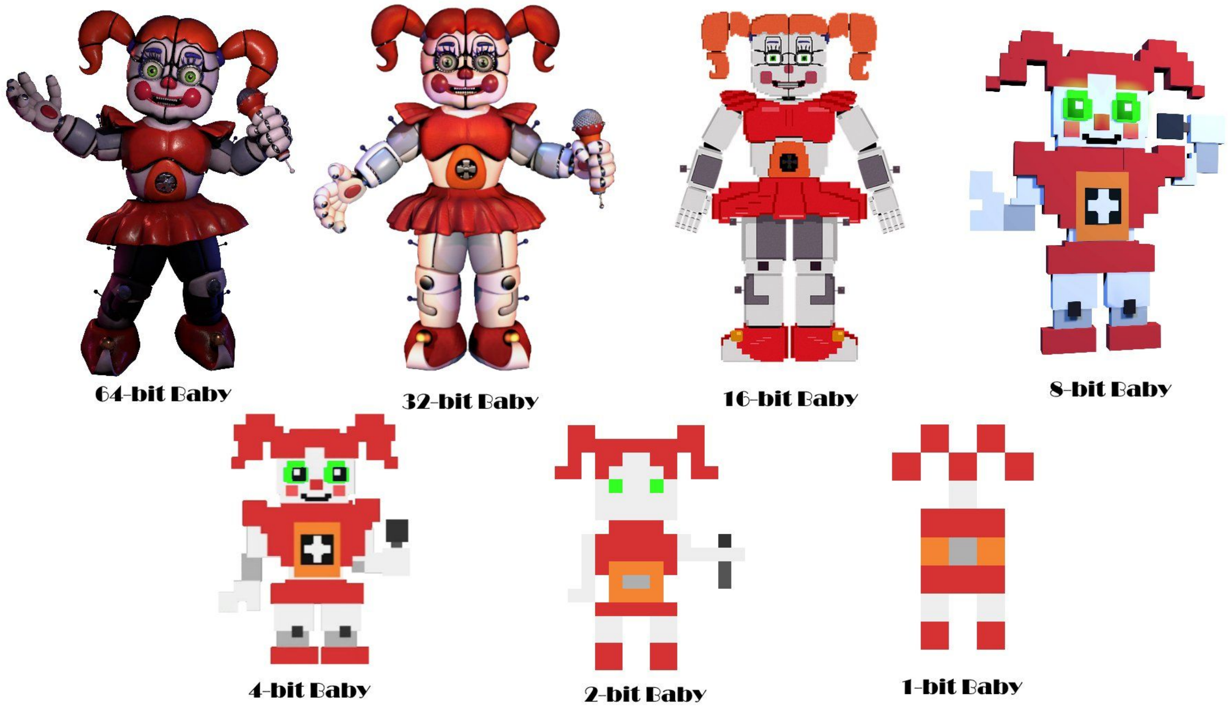
Bit depth





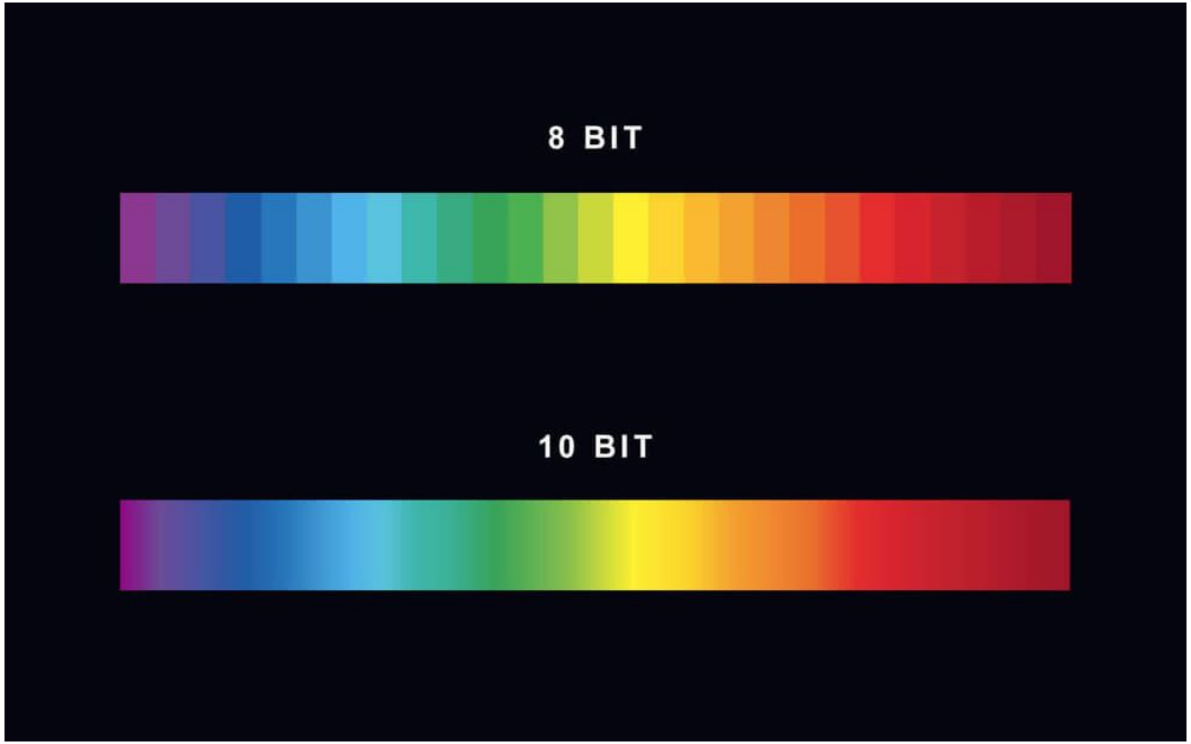
https://www.masteringbox.com/bit-depth/

Bit depth



https://twitter.com/mcaboyan/status/1352281438482272257

Bit depth

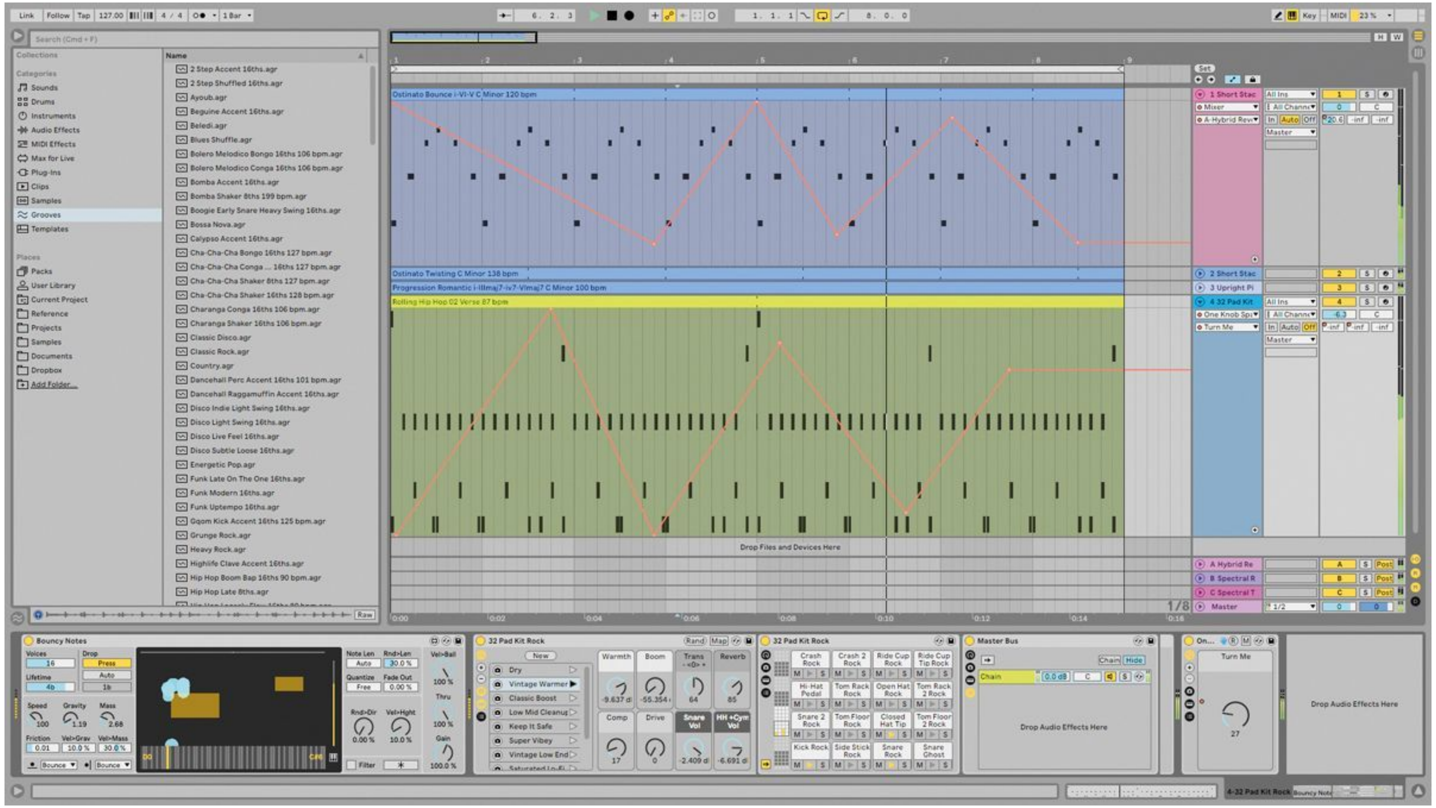


https://bit.ly/36QhF9y

Digital Music

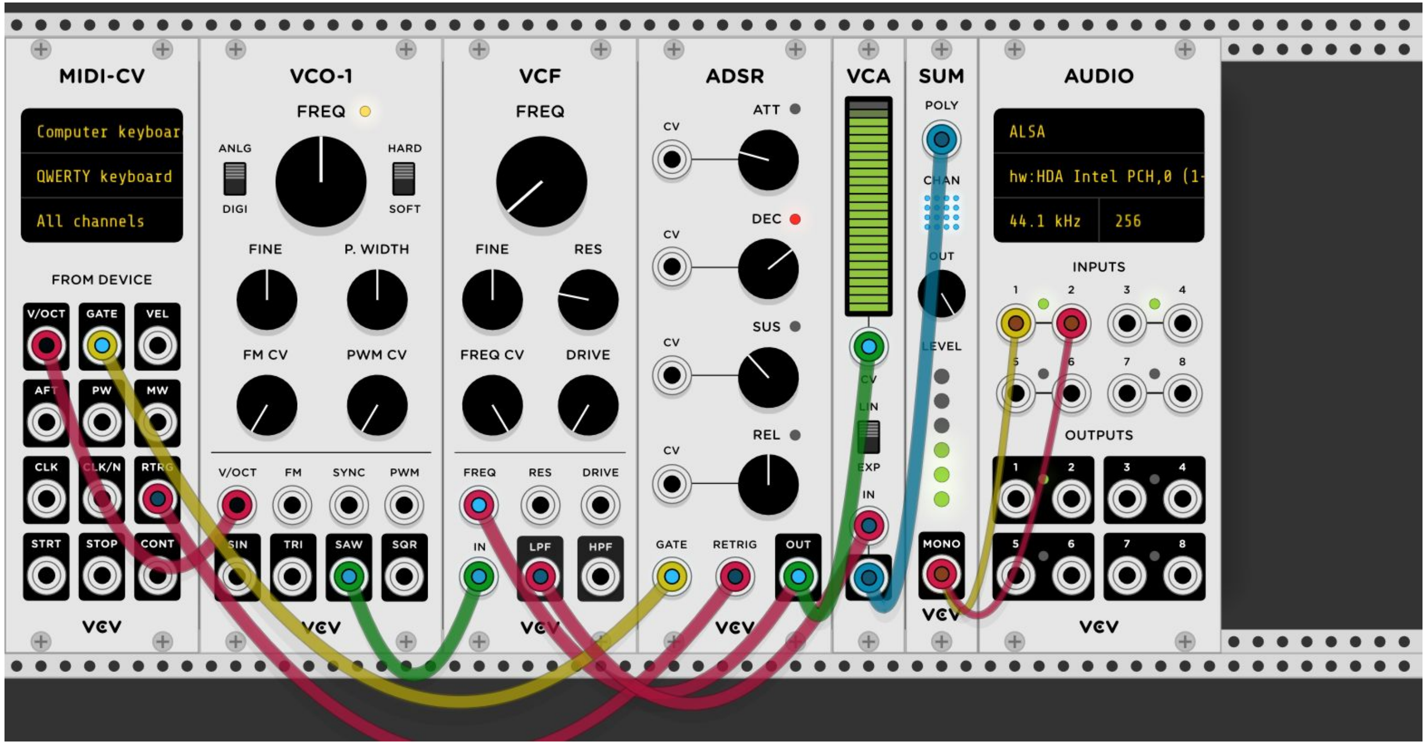
- Digital Audio Workstation (DAW)
- Plug-ins
- Standalone apps

Ableton Live 11



https://www.musicradar.com/news/ableton-live-11-suite

VCV Rack

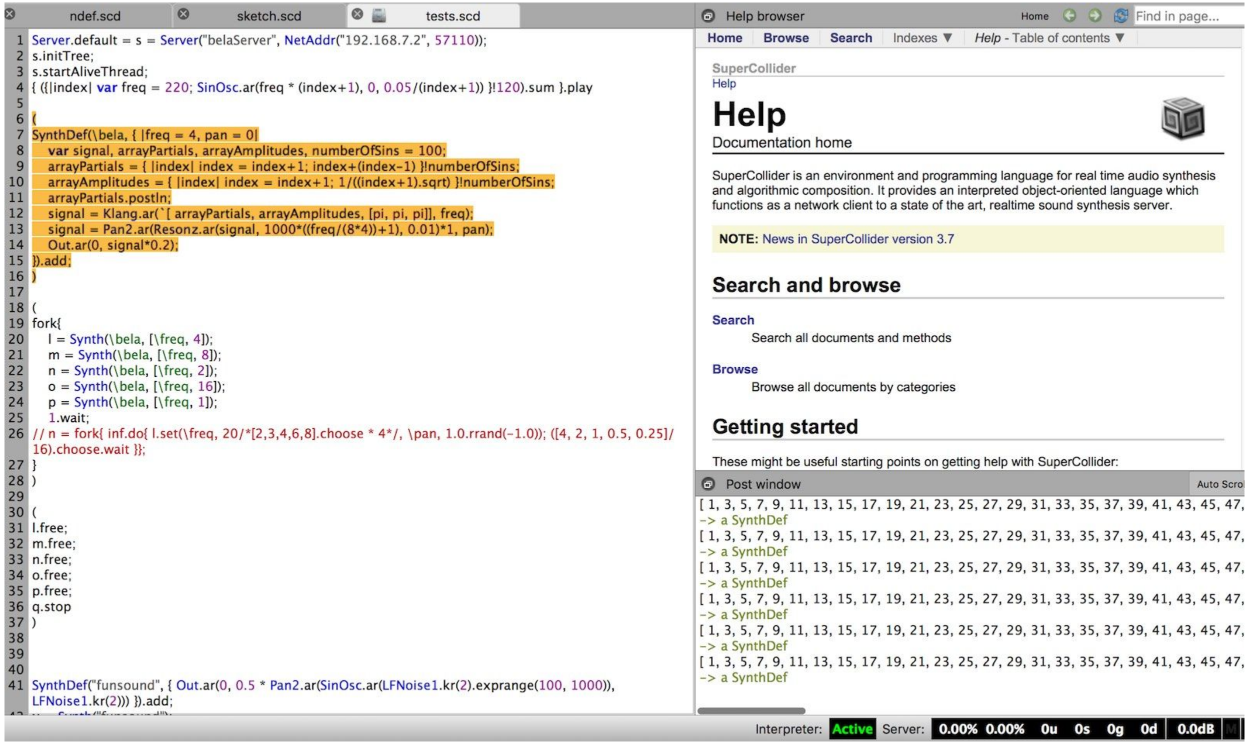


https://cdm.link/2019/02/vcv-rack-1-0/

Music programming

- Text-based
- Graphic

SuperCollider



```
Server.default = s = Server("belaServer", NetAddr("192.168.7.2", 57110));
s.initTree;
s.startAliveThread;
{ ({|index| var freq = 220; SinOsc.ar(freq * (index+1), 0, 0.05/(index+1)) }!120).sum }.play

(
SynthDef(\bela, { |freq = 4, pan = 0|
    var signal, arrayPartials, arrayAmplitudes, numberOfSins = 100;
    arrayPartials = { |index| index = index+1; index+(index-1) }!numberOfSins;
    arrayAmplitudes = { |index| index = index+1; 1/((index+1).sqrt) }!numberOfSins;
    arrayPartials.postln;
    signal = Klang.ar(`[ arrayPartials, arrayAmplitudes, [pi, pi, pi]], freq);
    signal = Pan2.ar(Resonz.ar(signal, 1000*((freq/(8*4))+1), 0.01)*1, pan);
    Out.ar(0, signal*0.2);
}).add;
)

(
fork{
    l = Synth(\bela, [\freq, 4]);
    m = Synth(\bela, [\freq, 8]);
    n = Synth(\bela, [\freq, 2]);
    o = Synth(\bela, [\freq, 16]);
    p = Synth(\bela, [\freq, 1]);
    1.wait;
// n = fork{ inf.do{ l.set(\freq, 20/*[2,3,4,6,8].choose * 4*/, \pan, 1.0.rrand(-1.0)); ([4, 2, 1, 0.5, 0.25]/
16).choose.wait }};
}
)

(
l.free;
m.free;
n.free;
o.free;
p.free;
q.stop
)



SynthDef("funsound", { Out.ar(0, 0.5 * Pan2.ar(SinOsc.ar(LFNoise1.kr(2).exprange(100, 1000)),
LFNoise1.kr(2))) }).add;
```

SuperCollider
Help

Help

Documentation home

SuperCollider is an environment and programming language for real time audio synthesis and algorithmic composition. It provides an interpreted object-oriented language which functions as a network client to a state of the art, realtime sound synthesis server.

NOTE: News in SuperCollider version 3.7

Search and browse

Search
Search all documents and methods

Browse
Browse all documents by categories

Getting started

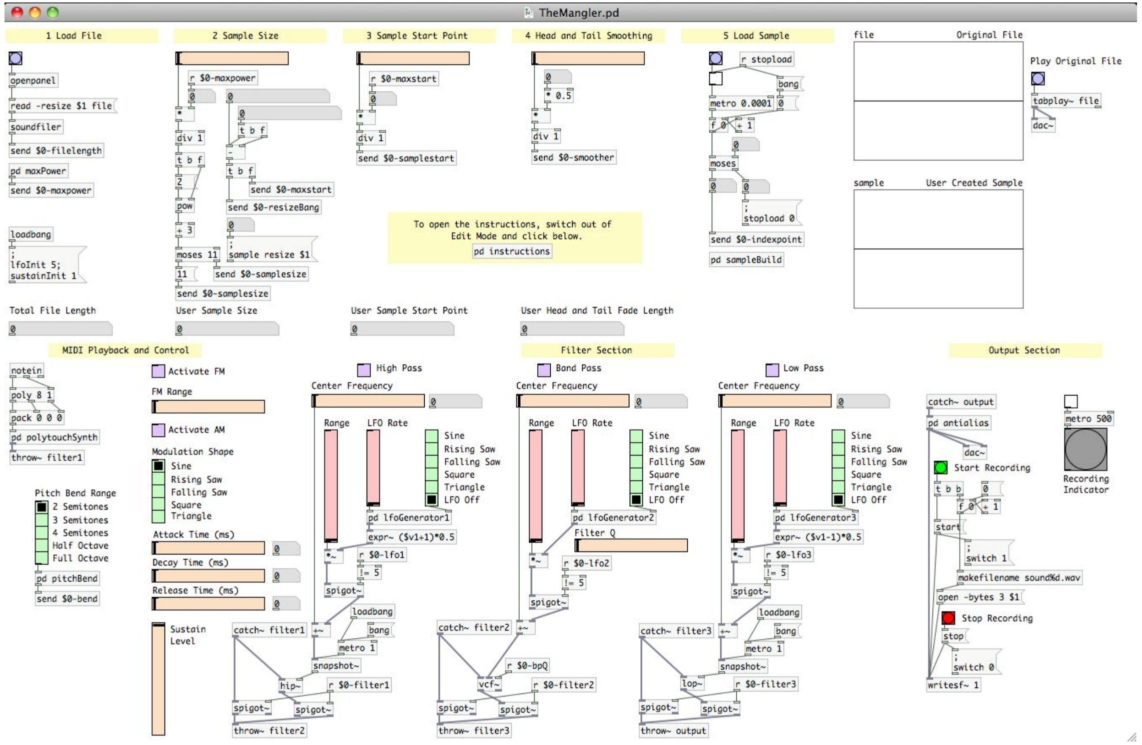
These might be useful starting points on getting help with SuperCollider:

```
[ 1, 3, 5, 7, 9, 11, 13, 15, 17, 19, 21, 23, 25, 27, 29, 31, 33, 35, 37, 39, 41, 43, 45, 47,
-> a SynthDef
[ 1, 3, 5, 7, 9, 11, 13, 15, 17, 19, 21, 23, 25, 27, 29, 31, 33, 35, 37, 39, 41, 43, 45, 47,
-> a SynthDef
[ 1, 3, 5, 7, 9, 11, 13, 15, 17, 19, 21, 23, 25, 27, 29, 31, 33, 35, 37, 39, 41, 43, 45, 47,
-> a SynthDef
[ 1, 3, 5, 7, 9, 11, 13, 15, 17, 19, 21, 23, 25, 27, 29, 31, 33, 35, 37, 39, 41, 43, 45, 47,
-> a SynthDef
[ 1, 3, 5, 7, 9, 11, 13, 15, 17, 19, 21, 23, 25, 27, 29, 31, 33, 35, 37, 39, 41, 43, 45, 47,
-> a SynthDef
[ 1, 3, 5, 7, 9, 11, 13, 15, 17, 19, 21, 23, 25, 27, 29, 31, 33, 35, 37, 39, 41, 43, 45, 47,
-> a SynthDef
```

Interpreter: Active Server: 0.00% 0.00% 0u 0s 0g 0d 0.0dB

https://blog.bela.io/live-coding-sensors-with-supercollider/

Pure Data



To open the instructions, switch out of Edit Mode and click below.

pd instructions

https://designingsound.org/2013/04/02/pure-data-wavetable-synth-part-1/

What are their advantages and disadvantages?

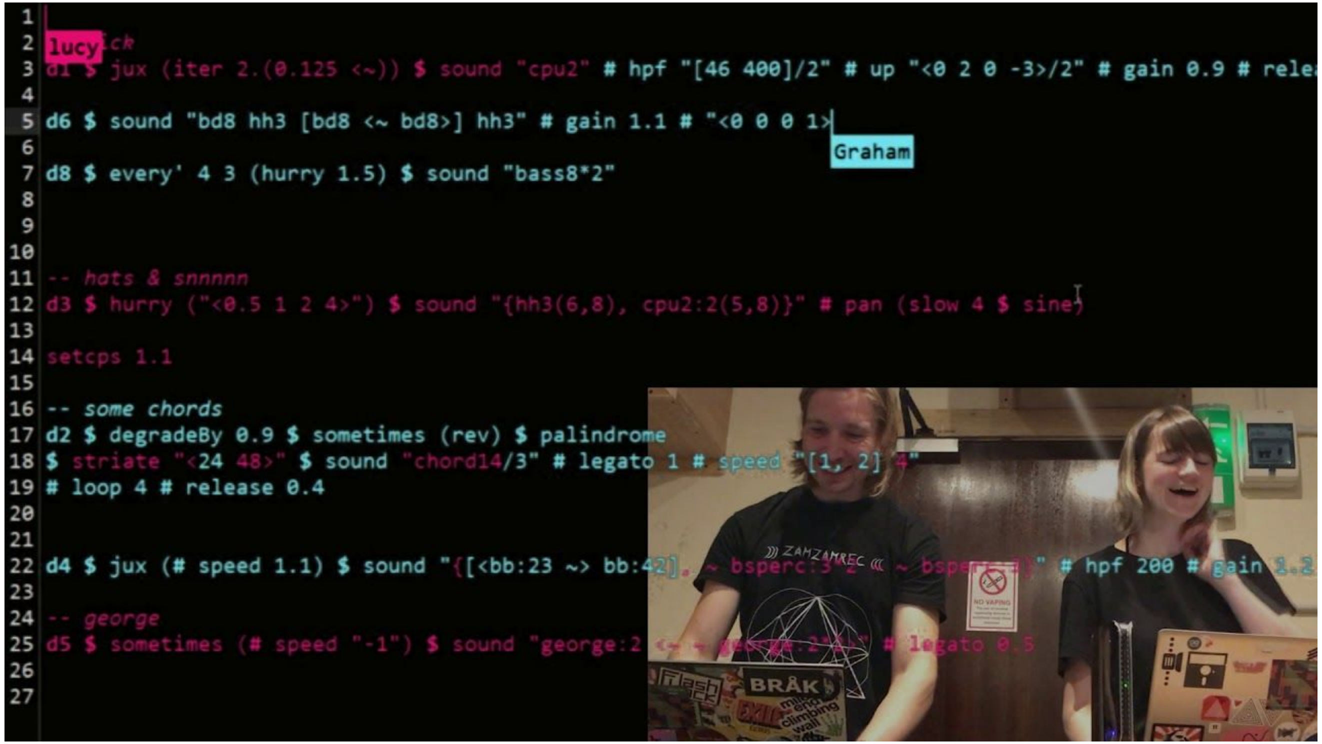
- Text-based
- Graphic

Live coding

- Interaction unit: bar
- SuperCollider-related
  - TidalCycles
  - Sonic Pi
- Browser-based
  - Gibber
  - Glicol
- Visuals
  - Hydra

https://github.com/toplap/awesome-livecoding#languages

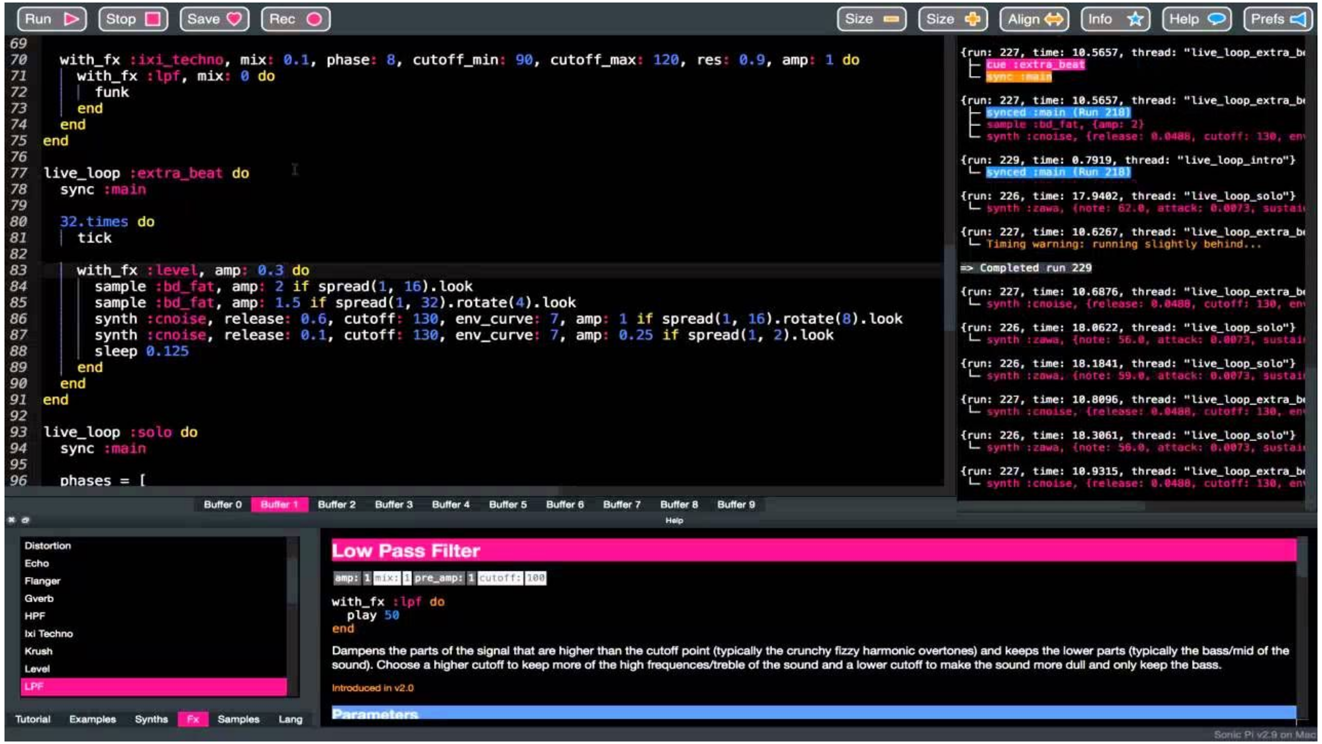
TidalCycles



```
lucy ck
d1 $ jux (iter 2.(0.125 <~)) $ sound "cpu2" # hpf "[46 400]/2" # up "<0 2 0 -3>/2" # gain 0.9 # rele

d6 $ sound "bd8 hh3 [bd8 <~ bd8>] hh3" # gain 1.1 # "<0 0 0 1>
                                                  Graham
d8 $ every' 4 3 (hurry 1.5) $ sound "bass8*2"



-- hats & snnnnn
d3 $ hurry ("<0.5 1 2 4>") $ sound "{hh3(6,8), cpu2:2(5,8)}" # pan (slow 4 $ sine)

setcps 1.1

-- some chords
d2 $ degradeBy 0.9 $ sometimes (rev) $ palindrome
$ striate "<24 48>" $ sound "chord14/3" # legato 1 # speed "[1, 2] 4"
# loop 4 # release 0.4


d4 $ jux (# speed 1.1) $ sound "{[<bb:23 ~> bb:42], ~ bsperc:3 2 ~ bsperc:2}" # hpf 200 # gain 1.2

-- george
d5 $ sometimes (# speed "-1") $ sound "george:2 <~ ~ george:2*2>" # legato 0.5
```

https://tidalcycles.org/

Sonic Pi



```

  with_fx :ixi_techno, mix: 0.1, phase: 8, cutoff_min: 90, cutoff_max: 120, res: 0.9, amp: 1 do
    with_fx :lpf, mix: 0 do
      funk
    end
  end
end

live_loop :extra_beat do
  sync :main

  32.times do
    tick

    with_fx :level, amp: 0.3 do
      sample :bd_fat, amp: 2 if spread(1, 16).look
      sample :bd_fat, amp: 1.5 if spread(1, 32).rotate(4).look
      synth :cnoise, release: 0.6, cutoff: 130, env_curve: 7, amp: 1 if spread(1, 16).rotate(8).look
      synth :cnoise, release: 0.1, cutoff: 130, env_curve: 7, amp: 0.25 if spread(1, 2).look
      sleep 0.125
    end
  end
end

live_loop :solo do
  sync :main

  phases = [
```

```
{run: 227, time: 10.5657, thread: "live_loop_extra_b
 ├─ cue :extra_beat
 └─ sync :main

{run: 227, time: 10.5657, thread: "live_loop_extra_b
 ├─ synced :main (Run 218)
 ├─ sample :bd_fat, {amp: 2}
 └─ synth :cnoise, {release: 0.0488, cutoff: 130, en

{run: 229, time: 0.7919, thread: "live_loop_intro"}
 └─ synced :main (Run 218)

{run: 226, time: 17.9402, thread: "live_loop_solo"}
 └─ synth :zawa, {note: 62.0, attack: 0.0073, sustai

{run: 227, time: 10.6267, thread: "live_loop_extra_b
 └─ Timing warning: running slightly behind...

=> Completed run 229

{run: 227, time: 10.6876, thread: "live_loop_extra_b
 └─ synth :cnoise, {release: 0.0488, cutoff: 130, en

{run: 226, time: 18.0622, thread: "live_loop_solo"}
 └─ synth :zawa, {note: 56.0, attack: 0.0073, sustai

{run: 226, time: 18.1841, thread: "live_loop_solo"}
 └─ synth :zawa, {note: 59.0, attack: 0.0073, sustai

{run: 227, time: 10.8096, thread: "live_loop_extra_b
 └─ synth :cnoise, {release: 0.0488, cutoff: 130, en

{run: 226, time: 18.3061, thread: "live_loop_solo"}
 └─ synth :zawa, {note: 56.0, attack: 0.0073, sustai

{run: 227, time: 10.9315, thread: "live_loop_extra_b
 └─ synth :cnoise, {release: 0.0488, cutoff: 130, en
```

Low Pass Filter

`amp: 1` `mix: 1` `pre_amp: 1` `cutoff: 100`

```
with_fx :lpf do
  play 50
end
```

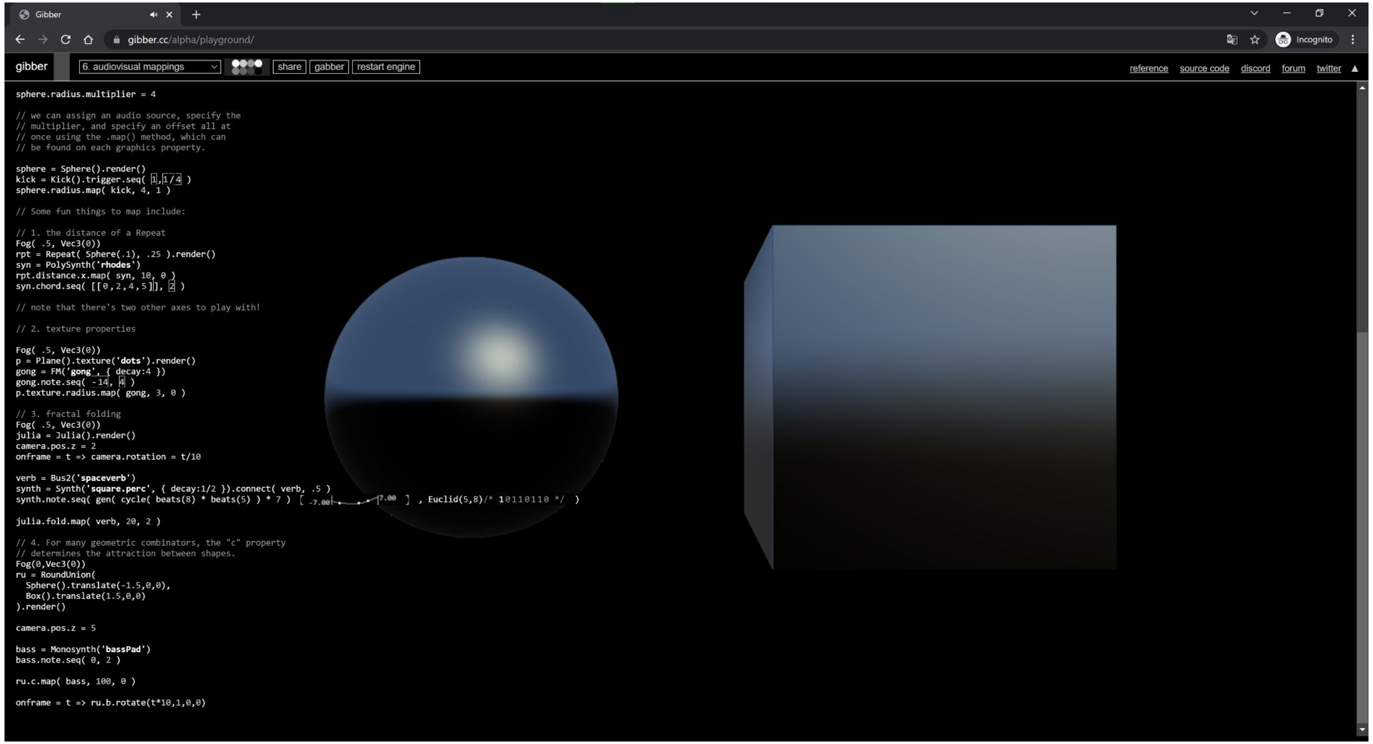
Dampens the parts of the signal that are higher than the cutoff point (typically the crunchy fizzy harmonic overtones) and keeps the lower parts (typically the bass/mid of the sound). Choose a higher cutoff to keep more of the high frequences/treble of the sound and a lower cutoff to make the sound more dull and only keep the bass.

Introduced in v2.0

Parameters

https://sonic-pi.net/

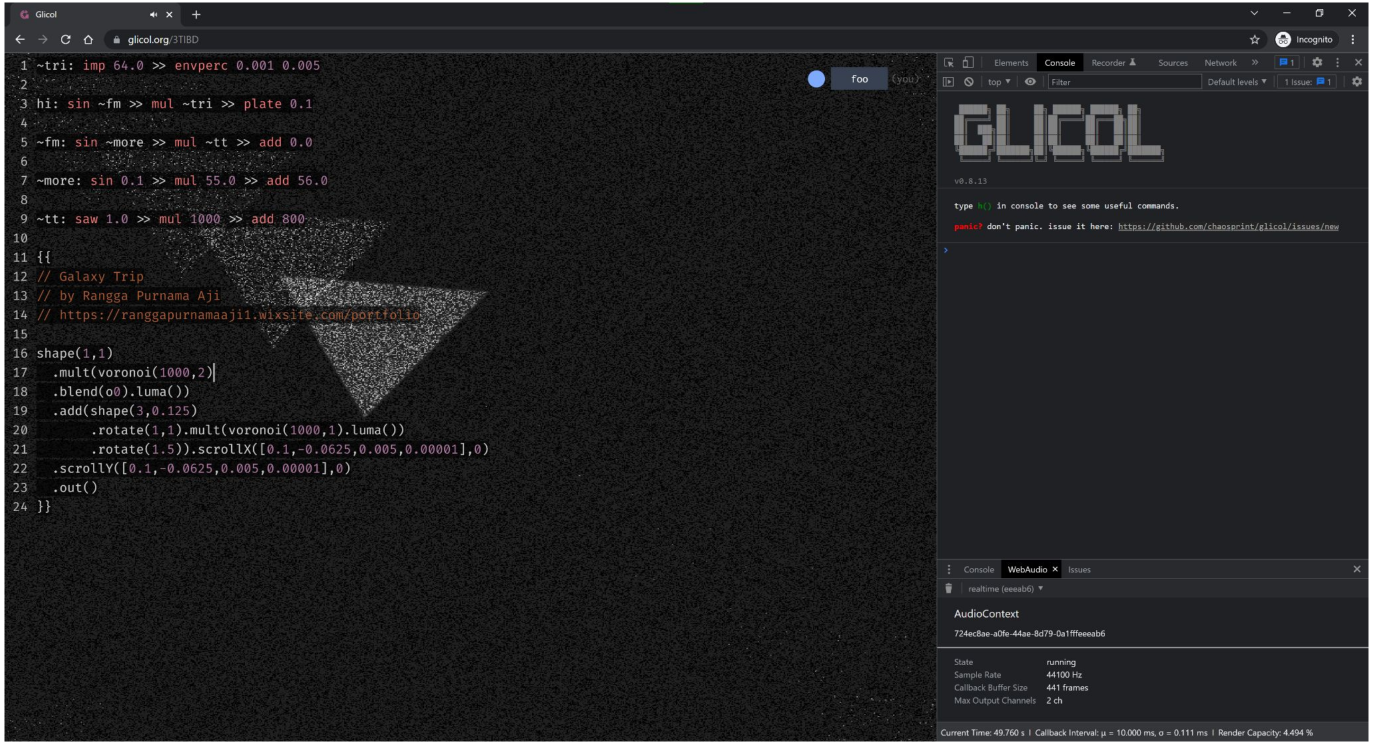
Gibber



```
sphere.radius.multiplier = 4

// we can assign an audio source, specify the
// multiplier, and specify an offset all at
// once using the .map() method, which can
// be found on each graphics property.

sphere = Sphere().render()
kick = Kick().trigger.seq( 1,1/4 )
sphere.radius.map( kick, 4, 1 )

// Some fun things to map include:

// 1. the distance of a Repeat
Fog( .5, Vec3(0))
rpt = Repeat( Sphere(.1), .25 ).render()
syn = PolySynth('rhodes')
rpt.distance.x.map( syn, 10, 0 )
syn.chord.seq( [[0,2,4,5]], 2 )

// note that there's two other axes to play with!

// 2. texture properties

Fog( .5, Vec3(0))
p = Plane().texture('dots').render()
gong = FM('gong', { decay:4 })
gong.note.seq( -14, 4 )
p.texture.radius.map( gong, 3, 0 )

// 3. fractal folding
Fog( .5, Vec3(0))
julia = Julia().render()
camera.pos.z = 2
onframe = t => camera.rotation = t/10

verb = Bus2('spaceverb')
synth = Synth('square.perc', { decay:1/2 }).connect( verb, .5 )
synth.note.seq( gen( cycle( beats(8) * beats(5) ) * 7 ) [ -7.00 7.00 ] , Euclid(5,8)/* 10110110 */  )

julia.fold.map( verb, 20, 2 )

// 4. For many geometric combinators, the "c" property
// determines the attraction between shapes.
Fog(0,Vec3(0))
ru = RoundUnion(
  Sphere().translate(-1.5,0,0),
  Box().translate(1.5,0,0)
).render()

camera.pos.z = 5

bass = Monosynth('bassPad')
bass.note.seq( 0, 2 )

ru.c.map( bass, 100, 0 )

onframe = t => ru.b.rotate(t*10,1,0,0)
```

https://gibber.cc/alpha/playground/

Glicol x Hydra



```
~tri: imp 64.0 >> envperc 0.001 0.005

hi: sin ~fm >> mul ~tri >> plate 0.1

~fm: sin ~more >> mul ~tt >> add 0.0

~more: sin 0.1 >> mul 55.0 >> add 56.0

~tt: saw 1.0 >> mul 1000 >> add 800

{{
// Galaxy Trip
// by Rangga Purnama Aji
// https://ranggapurnamaaji1.wixsite.com/portfolio

shape(1,1)
  .mult(voronoi(1000,2))
  .blend(o0).luma())
  .add(shape(3,0.125)
       .rotate(1,1).mult(voronoi(1000,1).luma())
       .rotate(1.5)).scrollX([0.1,-0.0625,0.005,0.00001],0)
  .scrollY([0.1,-0.0625,0.005,0.00001],0)
  .out()
}}
```

```
type h() in console to see some useful commands.
panic? don't panic. issue it here: https://github.com/chaosprint/glicol/issues/new
```

AudioContext

724ec8ae-a0fe-44ae-8d79-0a1fffeeeab6

```
State                  running
Sample Rate            44100 Hz
Callback Buffer Size   441 frames
Max Output Channels    2 ch
```

Current Time: 49.760 s | Callback Interval: μ = 10.000 ms, σ = 0.111 ms | Render Capacity: 4.494 %

https://glicol.org

Advantages for digital tools?

- Portability
- Accessibility
- Algorithm
- Collaboration