Making Music With Code

Russell Snyder
Digital Composition and Performance, MSc
Fullstack Developer/Consultant - Wunderdog GmbH

Overview

Why - escaping skeuomorphism, physically impossible sounds, getting out of yourself, process focus

What - creating digital music without a GUI - speaking directly to the computer

Who - algo-ravers, sound designers, composers

Where - online, studio, international, live avante-garde

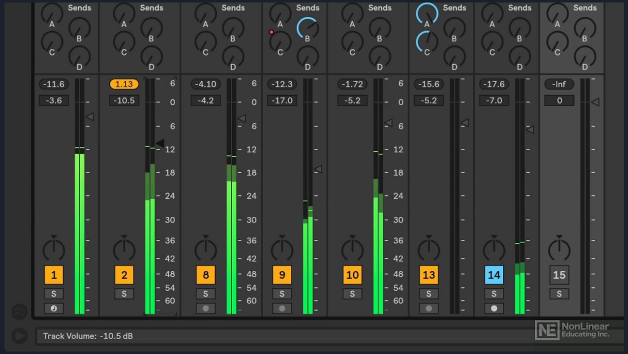
When - NOW!

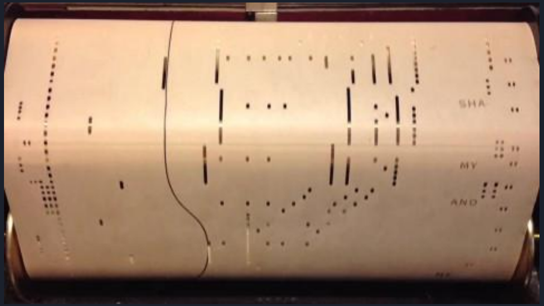
How - live examples with Supercollider

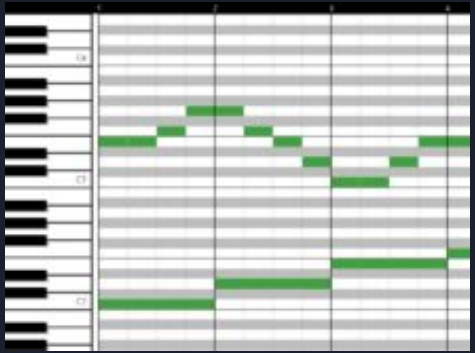
Why - Escaping Skeuomorphism

Skeuomorphism - graphical user interface design to describe interface objects that mimic their real-world counterparts in how they appear and/or how the user can interact with them.

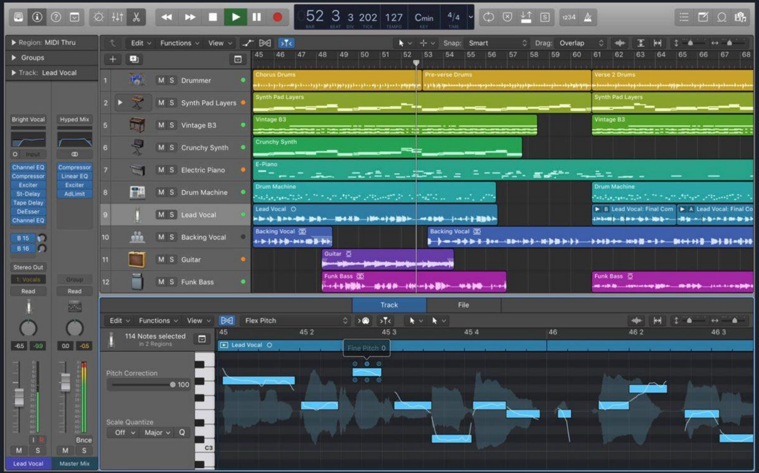
DAWs are build on earlier tech which is built on earlier tech which is based on earlier tech….

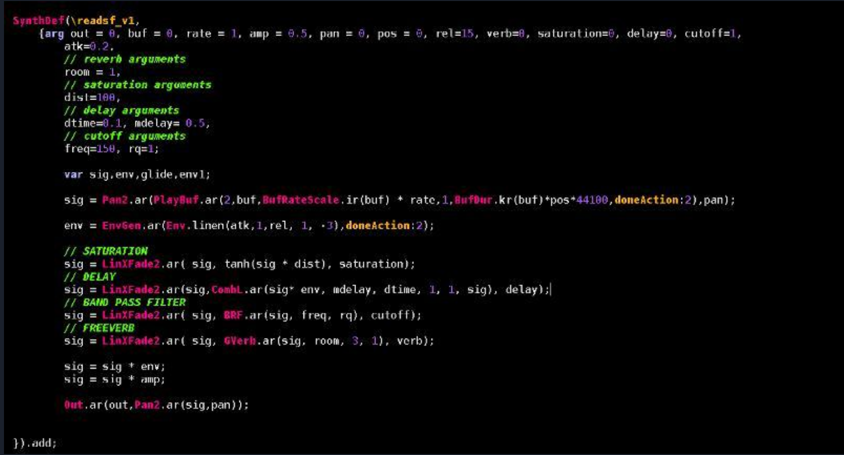






What - no pre-defined GUI





```
SynthDef(\readsf_v1,
    {arg out = 0, buf = 0, rate = 1, amp = 0.5, pan = 0, pos = 0, rel=15, verb=0, saturation=0, delay=0, cutoff=1,
        atk=0.2,
        // reverb arguments
        room = 1,
        // saturation arguments
        dist=100,
        // delay arguments
        dtime=0.1, mdelay= 0.5,
        // cutoff arguments
        freq=150, rq=1;

        var sig,env,glide,env1;

        sig = Pan2.ar(PlayBuf.ar(2,buf,BufRateScale.ir(buf) * rate,1,BufDur.kr(buf)*pos*44100,doneAction:2),pan);

        env = EnvGen.ar(Env.linen(atk,1,rel, 1, -3),doneAction:2);

        // SATURATION
        sig = LinXFade2.ar( sig, tanh(sig * dist), saturation);
        // DELAY
        sig = LinXFade2.ar(sig,CombL.ar(sig* env, mdelay, dtime, 1, 1, sig), delay);
        // BAND PASS FILTER
        sig = LinXFade2.ar( sig, BRF.ar(sig, freq, rq), cutoff);
        // FREEVERB
        sig = LinXFade2.ar( sig, GVerb.ar(sig, room, 3, 1), verb);

        sig = sig * env;
        sig = sig * amp;

        Out.ar(out,Pan2.ar(sig,pan));


}).add;
```

Traditional DAW workflow - Left to right, tracks, plugins - whole ecosystem

Code based workflow - start with nothing, build however you want

Get out of the box

A
Dm
A7
Dm
A7
Dm
B
D7
C7
F
A7
A7

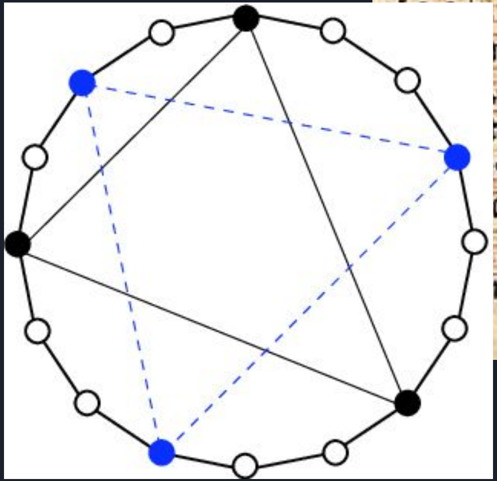
Metaphor constricts imagination

- Does sound go from left to right?

Prescribed workflow

Historically bound and ethnocentric

- Gregorian chant had no rhythm
- Not psychologically representative to our perception
- https://www.amazon.de/Geometry-Musical-Rhythm-Godfried-Toussaint/dp/1466512024



SO WHAT?!

Live Coding - why is it important?

“...being digital allows the process, not just the product, to be conveyed. That process can be the fantasy and ecstasy of one mind, or it can be the collective imagination of many, or it can be the vision of a revolutionary group.”

- p.232, BEING DIGITAL, Nicholas Negroponte

Shift from product to process happening everywhere

- organic foods
- Sustainability in products
- Transparency
- Silo to CI/CD Microservices

Nothing is Impossible!

Impossible to play: https://youtu.be/f2gVhBxwRqg?t=28

Impossible to enter by hand - black midi: https://youtu.be/tfrIXiemYqo?t=74

Impossible to compose without computer - statistical gas models: https://youtu.be/RC3XCfDBIK8?t=6

Impossible to think of - Markov Chains - https://www.youtube.com/watch?v=Ghqchw9X6dw

Impossible analog - sample level manipulation - Granulation: https://www.youtube.com/watch?v=flq-6qGYJP8&feature=youtu.be&t=174

Ai - https://magenta.tensorflow.org/

Who: Algo Ravers

```
///////ROBO MELODY
~robo1.play
~robo1=Pbind(\instrument, \robo1, \dur, Pseq([0.25, 0.5, 0.2
0.1, 0.05, 0.1, 0.0, 0.05]/2, inf))
~robo1.release

~snare.relea            elease;~robo1.release
```

Like a normal rave, but with Live coding

Often with visual components

Photo from Detroit area: https://www.vectorform.com/news-and-views/algorave-with-us/

Sound Design

Behind the scenes in almost every production

Procedural audio - modeling real world sounds

Difficult to get sounds - birds don’t sing the same in isolation

Great examples: http://aspress.co.uk/sd/practical28.html

https://en.wikibooks.org/wiki/Designing_Sound_in_SuperCollider/

Bird Specific:
https://web.archive.org/web/20080307143015/http://obiwannabe.co.uk/tutorials/html/tutorial_birds.html

Composers

Searching for new sounds, new ways to organize sound

Various supercollider composers: https://soundcloud.com/supercollider-x

Pushing boundaries with new technology!

Where: Online

Tone.js: https://tonejs.github.io/

Klangmeister: http://ctford.github.io/klangmeister/

Follow Web Audio Weekly: https://www.webaudioweekly.com/

Glitch:
https://naivesound.com/glitch/#%23%0A%23%20Kraftwerk%20-%20Das%20Model%0A%23%0A%0Abpm%3D120%0A%0Aparts%20%3D%20seq(bpm%2F16%2C%200%2C%201%2C%203%2C%207%2C%207%2C%207%2C%207%2C%203%2C%201)%0A%0Ai%20%3D%20seq(bpm%2F4%2C%200%2C%207)%0A%0Aroot%20%3D%20saw(hz(i))%0Athird%20%3D%20tri(hz(i%2B4))%0Afifth%20%3D%20saw(hz(i%2B7))%0A%0Achords%20%3D%20env(seq(bpm*2%2C1)*mix(%0A%20%20(0.5%2Croot)%0A%20%20(0.5%2Cthird)%0A%20%20(0.5%2Cfifth)%0A)%2C0.01%2C%200.1)%0A%0Asolo%20%3D%20seq(bpm*2%2C%200%2C%200%2C%200%2C%200%2C%203%2C%200%2C%203%2C%200%2C%20-5%2C%20-5%2C%20-5%2C%20-5%2C%20-2%2C%20-5%2C%20-2%2C%20-5)%0Asolo%20%3D%20fm(hz(solo-12)%2C%200.5%2C%201)%0A%0Adrums%20%3D%20tr808(seq(bpm*4%2C0%2C8%2C1%2C8)

Everywhere, but also Live in Berlin

Noise Fabrik https://noizefabrik.com/events/algorave-musik-hackspace/

Common Ground: https://commonground.community/

Loophole: https://www.facebook.com/berlin.loophole/

Google it ;-)

How: Supercollider

Free, Open source - https://github.com/supercollider/supercollider

Last major version was completely community released

Fast and efficient - built on C++11

Plug and play unit generators (UGens) and lots of extensions (Quarks)

Great documentation

Extensive pattern generation library (important for algorithmic composition)

Built solely to make sound/music

Sounds really good

What’s Special about Supercollider?

Architecture - Split client (sclang) from server (scsynth),

- Best practice to keep audio thread split from UI/Control and allows for various client languages to be built on top

Clojure - Overtone: http://overtone.github.io/

Python - FoxDot: https://foxdot.org/

Haskell - Tidal: https://tidalcycles.org/

Node - Supercollider.js https://crucialfelix.github.io/supercolliderjs/

Functional is favored for live-coding - more expressive

Well use ‘pure’ Supercollider at the start

Ugen Examples

Control rate vs Audio rate (.kr vs .ar)

TURN YOUR VOLUME DOWN!

Intro to IDE - running and stopping

Intro to using Documenation

LET’S MAKE SOME NOISE!

Pattern Example - basic

Algorithmic Composition specific Pattern library

Guide: http://doc.sccode.org/Tutorials/A-Practical-Guide/PG_01_Introduction.html

Basic Event example - Mention failed flowkey application…..

```
(
p = Pbind(
   \degree, Pseq(#[0, 0, 4, 4, 5, 5, 4], 1),
   \dur, Pseq(#[0.5, 0.5, 0.5, 0.5, 0.5, 0.5, 1], 1)
).asStream;    // remember, you have to make a stream out of the pattern before using it
)

p.next(())
```

Pattern Examples - Moderate

Occur in both sound design and compositions

From http://danielnouri.org/docs/SuperColliderHelp/Collections/SequenceableCollection.html

```
Array.series(5, 10, 2);

Array.geom(5, 1, 3);

Array.fib(5);

Array.fib(5, 2, 32); // start from 32 with step 2.

[1, 2, 3, 4].wchoose([0.1, 0.2, 0.3, 0.4]);
```

Pattern Examples - Advanced Composition and JIT Changes

```
~rhythm = EventPatternProxy(Pbind(
   \dur, Pwrand(#[0.125, 0.25, 0.5], #[0.3, 0.5, 0.2], inf),
   \legato, Pwrand(#[0.1, 0.6, 1.01], #[0.1, 0.3, 0.6], inf)
));

~melody = EventPatternProxy(Pbind(
   \degree, Pwhite(-4, 11, inf)
));

p = Pchain(~melody, ~rhythm).play;

~melody.source = PmonoArtic(\default, \degree, Pseries(4, Prand(#[-1, 1], inf), inf).fold(-4, 11));

~melody.source = Pbind(\degree, Pseries(4, Pwrand(#[-2, -1, 1, 2], #[0.3, 0.2, 0.2, 0.3], inf),
inf).fold(-4, 11));

p.stop;
```

Supercollider Birds Project

Demo of Bird Synth Project

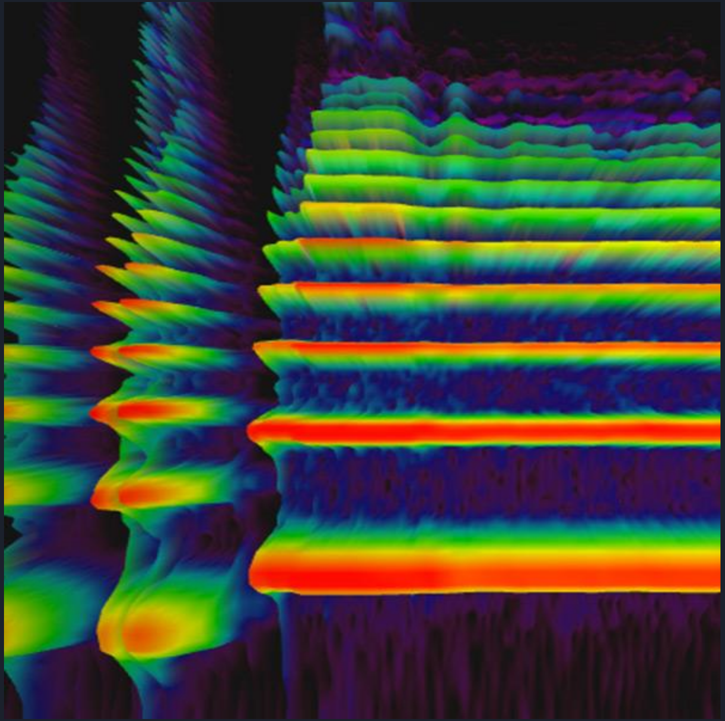
Resynthesis spectrograms of bird calls

Based on Common Lisp Music examples

Github page:

https://github.com/RussellSnyder/Supercollider-BirdSongs

Ex. Warbler, Oriole, CassinsKingbird



Sound Design: Bubbles - Sound

```
// sound of a bubble - From Andy Farnell (I think)
(
SynthDef(\bubblebub, { |out=0, t_trig=0, attack=0.01, decay=0.08,
pitchcurvelen=0.1, freq=500, freq_spread=50 doneAction=0, amp=0.1|
     var pitch, son;
     amp   = amp * EnvGen.ar(Env.perc(attack, decay).delay(0.003), t_trig,
doneAction: doneAction);
     pitch = rrand(freq - freq_spread, freq + freq_spread) *
EnvGen.ar(Env.new([0,0,1],[0,1]).exprange(1, 2.718), t_trig, timeScale:
pitchcurvelen);
     son = SinOsc.ar(pitch);
     // high-pass to remove any lowpitched artifacts, scale amplitude
     son = HPF.ar(son, 500) * amp * 10;
     Out.ar(out, son);
}).store
)

x = Synth(\bubblebub);
x.set(\t_trig, 1);
x.free;
```

Sound Design: Bubbles - Sequence

```
// Sequence
(
p = Pbind(
    \instrument, \bubblebub,
    \sizefactor, Pwhite(0.0,1,inf),
    \dur, Pgauss(0.3, 0.2),
    \freq,  Pkey(\sizefactor).linexp(0, 1, 500, 900),
    \amp ,  Pkey(\sizefactor).linlin(0, 1, 0.15, 0.04),
    \decay, Pkey(\sizefactor).linlin(0, 1, 0.05, 0.08),
    \doneAction, 2
).play
)
p.stop
```

Compositions - Tweet Sized
@sc140tweets, #sc140, #supercollider

```
{|i|x=i+6.rand;Pbind(\dur,0.06,\sustain,1,\amp,0.01,\degree,Pgauss(x,sin(x+Ptime()%6/6e3)*9),\pan,Pkey(\degree)-x*9).play}!6//#SuperCollider
```

```
play{GVerb.ar(IFFT(PV_BrickWall(FFT(Buffer.alloc(s,1024),WhiteNoise.ar*Pulse.ar(8,2e-2)),SinOsc.ar(Duty.kr(1,0,Dseq((10..19),inf))))))}
```

```
play{a=SinOsc;p=Pulse;WhiteNoise.ar*p.kr(8,0.01)+a.ar(98*n=p.ar(4),0,p.kr(2,add:1))+GVerb.ar(a.ar(99*n)+p.ar(p.kr(3*n)),1,mul:0.1)}// #sc140
```

Interactive Examples with Mouse

```
// Intro to Documentation

// Gendy - what Xenakis hypothesized using punch cards….3 versions in Supercollider

(
    {
      Pan2.ar(
        CombN.ar(
          Resonz.ar(
            Gendy1.ar(2, 3, minfreq:1, maxfreq: MouseX.kr(10, 700), durscale:0.1, initCPs:10),
            MouseY.kr(50, 1000), 0.1),
          0.1, 0.1, 5, 0.6
        ),
        0.0)
    }.play
  )
```

Compositions - Generative / Phat Beats

Note about Quarks / extensions of Supercollider

Generative demos from ExampleCompositions.scd

Phat Beat demo from beat180805.scd

Live Coding Demos

Truly great tutorial:

https://github.com/theseanco/howto_co34pt_liveCode

GO MAKE SOME NOISE!