

Welcome to Sonic Pi

소닉파이 다운로드

<https://sonic-pi.net/>

오늘 강의 자료

<https://github.com/byulparan/acc-sonicpi>

인공지능과 전자음악

DAY2. 전자예술의 경험

전자음악 초기의 전자악기로, 소리를 만들어 내는 하나의 방법인 사운드 씬서시스는 미디어테크놀로지의 하나로 수학, 물리학까지 활용되는 융복합적 기술. 전자악기와 워크숍 소프트웨어를 이용하여 간단한 미디어아트 음악을 만들어 본다.

박성민



- 2021 뉴바로크 컴퍼니 [세계의 조화] 공연 전자음악 작곡 @세종문화회관 S씨어터
- 2020 현대자동차 제로원 랩 Z-LAB 크리에이터
- 2019 현대자동차 제로원 크리에이터(ZERO1NE Creator) 선정
- 2019 퍼퓸2019 : 린킨아웃 참여 @일민미술관
- 2019 SOFTWARE EDU FEST 2019 오프닝 공연 @성수동 어반소스

전자음악과 컴퓨터



근래들어 가요 및 힙합 프로듀서들이 대중적 인기를 얻으면서 그들의 작업실 및 작업 환경이 미디어에 많이 노출되고 있습니다.



장르에 무관하게 음악 창작에 있어 컴퓨터는 전방위 적으로 활용되고 있습니다.

<https://www.facebook.com/AOMGOFFICIAL/videos/1878061155741245/>



개인의
시간

두 번째 하이에나 그레이

KBS2

일반적으로 음악을 창작하기 위한 소프트웨어들을 DAW(Digital Audio Workstation) 라고 이야기합니다.



Apple Logic



Ableton Live



Steinberg Cubase

음악 소프트웨어의 역사

1983년 MIDI(Musical Instrument Digital Interface)



미디 시퀀서 프로그램들의 등장

전자악기들을 어떻게 효과적으로 제어할 것인가?



90년대 중.후반 컴퓨터로 오디오 레코딩 및 편집, 믹스등이
가능해지면서 DAW 라 불리우게 됨

그전에 이미



- MUSIC was developed by Mathews on an IBM 704 at Bell Labs in 1957^[3] (this original version was later referred to as MUSIC I)
- MUSIC II was developed by Mathews on an IBM 7094 at Bell Labs in 1958^[3]
- MUSIC III was developed by Mathews on an IBM 7090 at Bell Labs in 1960^[4]
- MUSIC IV was developed by Mathews and J. Miller on an IBM 7094 at Bell Labs in 1963^[4]

1950년 중반 Bell 연구소(Bell Telephone Laboratories)에서는 음성신호를 전화로 효율적으로 송수신하기 위해서 소리를 디지털화 하고 다시 재생하는 연구가 있었는데 연구진은 이를 통해 방송용 음악정보도 디지털화 할 수 도 있다고 생각했다.

당시 벨 연구소 연구원이었던 Mathews는 이런 분위기 속에서 컴퓨터로 소리를 합성하고 재생 할 수 있는 음악 언어 Music 1을 만들었다.

- 발췌

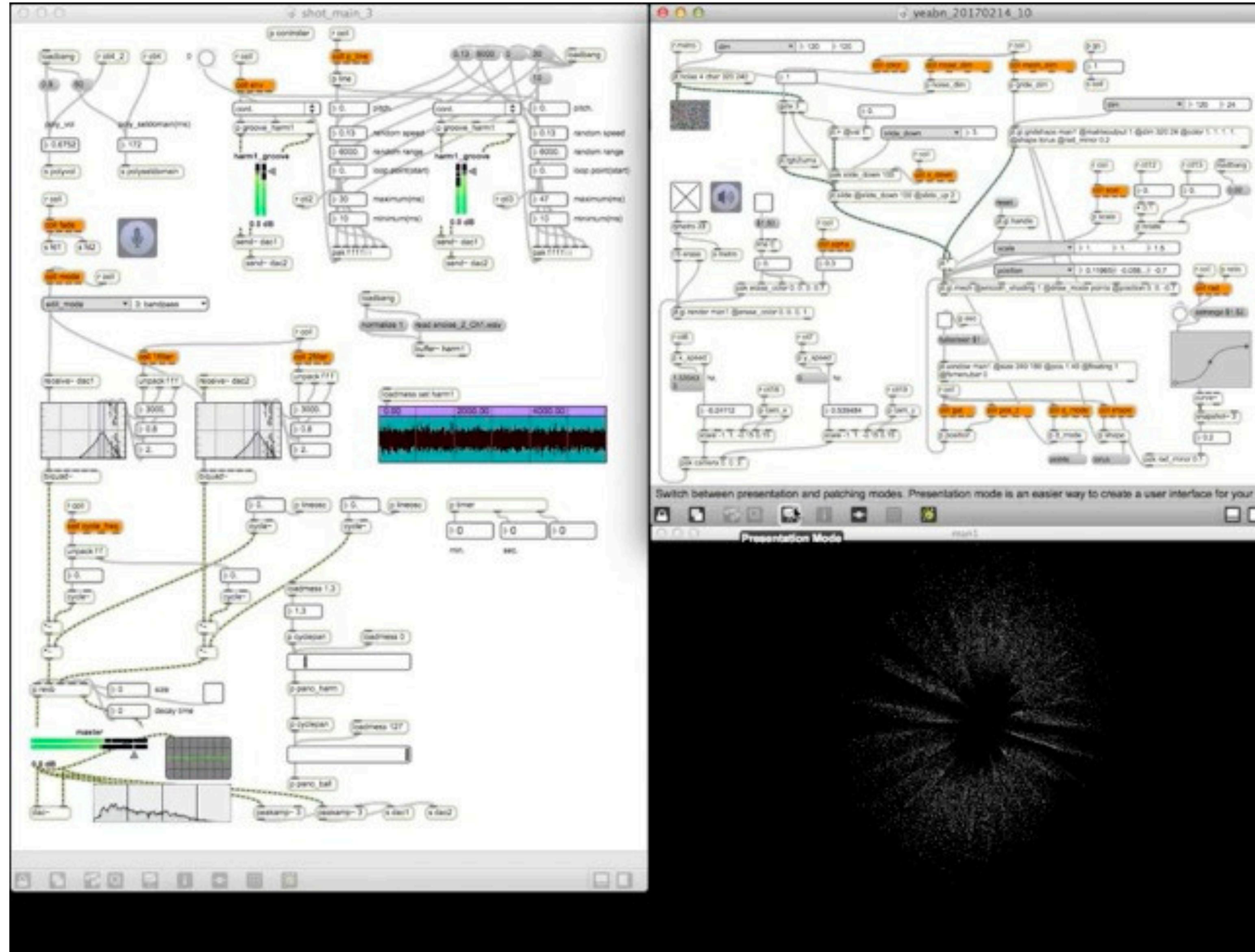
한국전자음악협회 학술지 <에밀레> 4호, 최수환 저 컴퓨터음악 언어의 역사

컴퓨터 음악의 아버지

Max Mathews



컴퓨터 음악을 위한 언어들(프로그래밍 도구)



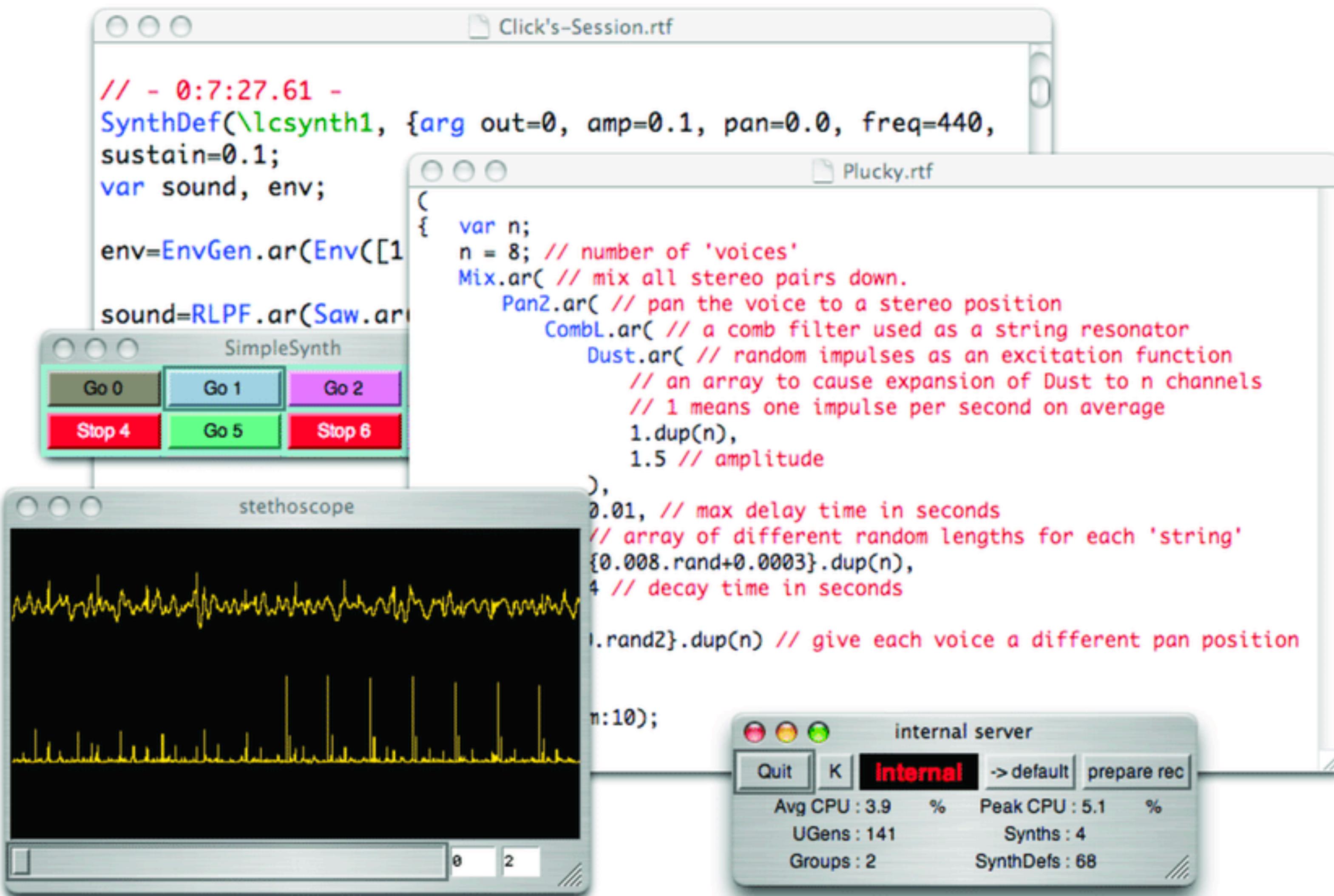
Max/MSP

마우스로 오브젝트들을 선으로 연결하는 방식으로 프로그래밍하는 미디어 아트계의 대표적인 프로그램.

사운드 / 그래픽스 등 거의 모든 미디어 매체들을 다룰 수 있다.

지금은 Ableton에 인수되었으면 Max4Live를 통해 Ableton Live 프로그램의 가능성을 무한하게 확장 가능하다.

컴퓨터 음악을 위한 언어들(프로그래밍 도구)



SuperCollider

현재 사운드 프로그래밍에서 가장 많이 사용되는 언어로
오픈소스 기반이며 강력하며 유연하고 넓은 확장성으로
꽤 넓게 활용된다.

우리가 오늘 공부할 SonicPI 도 내부적으로 SuperCollider 를
사운드 엔진으로 사용한다.

컴퓨터 음악의 두 가지 용도

1. 소리 합성(Digital Synthesis)

2. 음악적 구조 / 알고리즘 작곡

컴퓨터 음악 언어가 DAW 와 구분되는 점

-> 소리와 구조, 음악적 데이터를 실시간으로
생성(Generative)





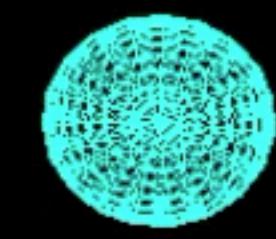
SK 이노베이션

CYMATICS ART by SILO LAB

Process:

Input text:

Natural Language Processing
삼겹살에 스즈가 무자게 땡긴다



Sonic PI : 소닉파이



Sonic Pi 는 원래 학교에서 컴퓨팅과 음악 수업을 모두 지원하도록 설계된 Ruby를 기반으로 하는 라이브 코딩 환경으로, 캠브리지 대학 컴퓨터 연구소 [1] 의 Sam Aaron 이 Raspberry Pi Foundation 과 공동으로 개발했습니다.

어렵고 사용하기 힘든 SuperCollider 를 교육용으로 활용하기 위해 쉬운 문법 + 복잡하지 않은 설정 + 잘 정리된 문서 등이 제공된다.

Sonic PI : 소닉파이

Sleep 과 리듬



Sonic PI : 소닉파이

코드(Chord)



C major: 도, 미, 솔

D minor: 레, 파, 라

E minor: 미, 솔, 시

F major: 파, 라, 도

G major: 솔, 시, 레

A minor: 라, 도, 미

Sonic PI : 소닉파이

스케일(scale):

음계



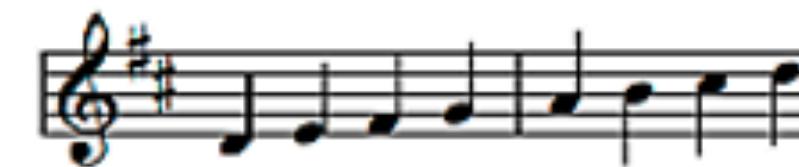
G Major

F Major



D Major

Bb Major



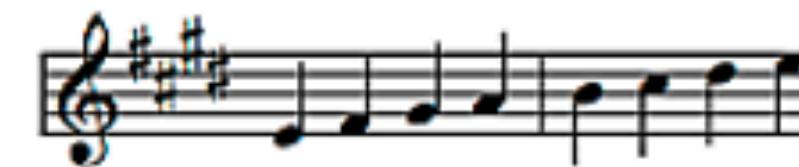
A Major

Eb Major



E Major

Ab Major



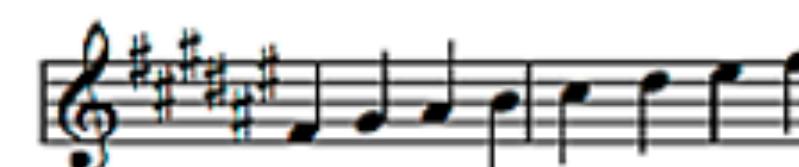
B Major

Db Major



F# Major

Gb Major



C# Major

Cb Major



Sonic PI : 소닉파이

쓰레드(thread)

```
1  
2 play :c4  
3 sleep 1  
4 play :g4  
5 sleep 1  
6 play :c5  
7 sleep 1  
8 play :g5  
9 sleep 1  
10  
11 loop do  
12     play :c3  
13     sleep 1  
14 end  
15  
16 loop do  
17     play :g4  
18     sleep 1  
19 end  
20  
21
```

프로그램의 흐름은 위에서부터 아래로
순차적으로 실행

여기에서 무한 루프 진입

이 지점으로 진행되지 못함

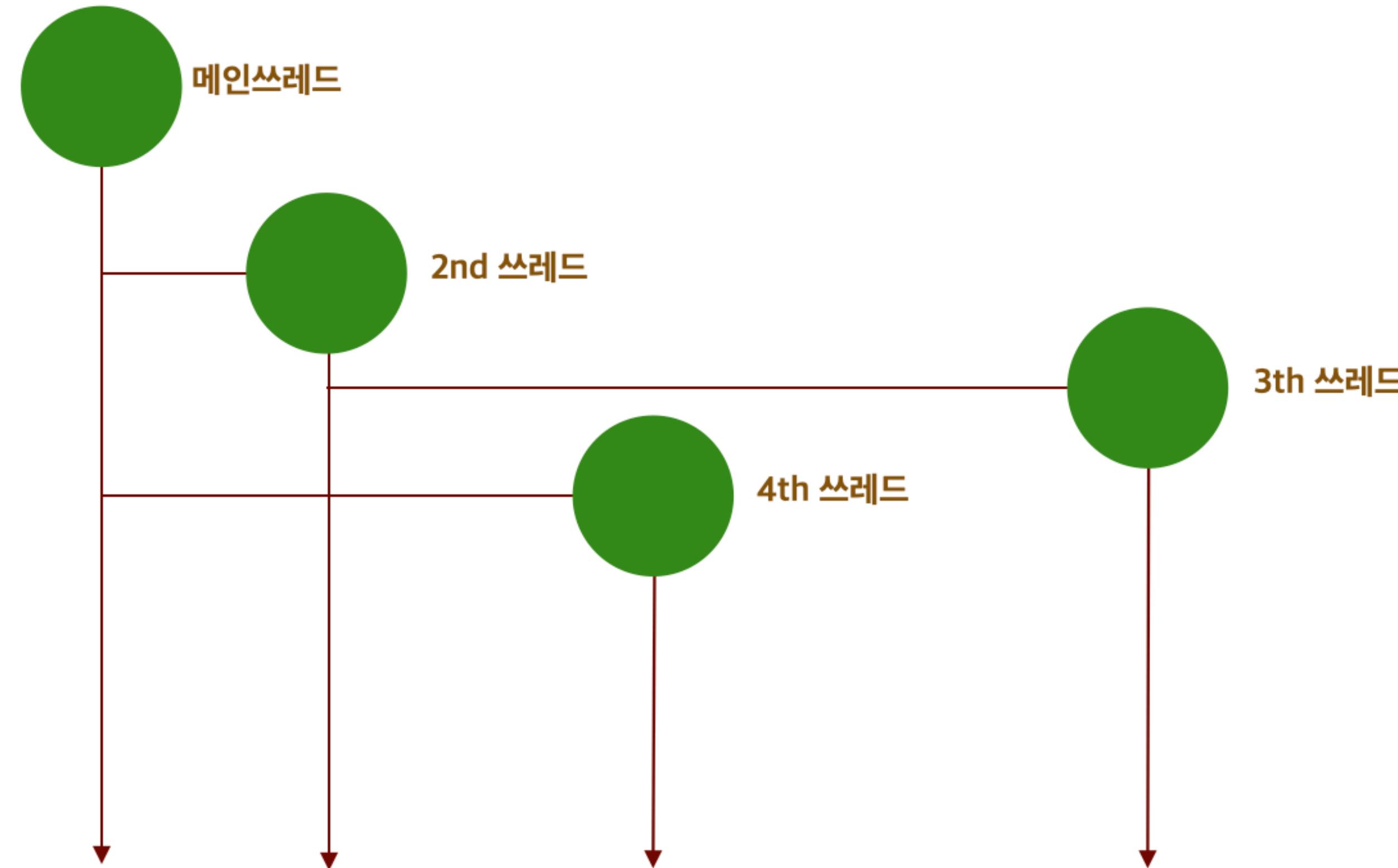
Sonic PI : 소닉파이

쓰레드(thread)

멀티 스레드(multi thread)

일반적으로 하나의 프로세스는 하나의 스레드를 가지고 작업을 수행하게 됩니다.

하지만 멀티 스레드(multi thread)란 하나의 프로세스 내에서 둘 이상의 스레드가 동시에 작업을 수행하는 것을 의미합니다.



Sonic PI : 소닉파이

쓰레드(thread)

```
2 # 쓰레드
3
4 play :c4
5 sleep 1
6 play :g4
7 sleep 1
8 play :c5
9 sleep 1
10 play :g5
11 sleep 1
12
13
14
15 in_thread do # thread
16   loop do
17     | play :c3
18     | sleep 1
19   end
20 end
21
22
23 loop do
24   play :g4
25   sleep 1
26 end
27
```

The diagram illustrates the execution flow of the Sonic Pi code. A vertical red line divides the code into two main sections. The top section contains a loop that runs once, followed by a nested loop that runs twice. The bottom section contains a loop that runs twice. Red arrows point from the start of each loop's body to its matching 'end' keyword, indicating the scope of each loop.