

**모두를 위한 SW 지식플러스**

2023년 10월 20일

충남대학교

**음악프로그래밍**

한양대학교 ERICA

소프트웨어융합대학 컴퓨터학부

도경구

음악 프로그래밍 = 소리 프로그래밍  
music sound



## 소리의 특성

- 물체가 진동(oscillation)하면서 공기에 파형을 만들어 소리가 나는데, 이를 **음파(sound wave)**라고 한다.



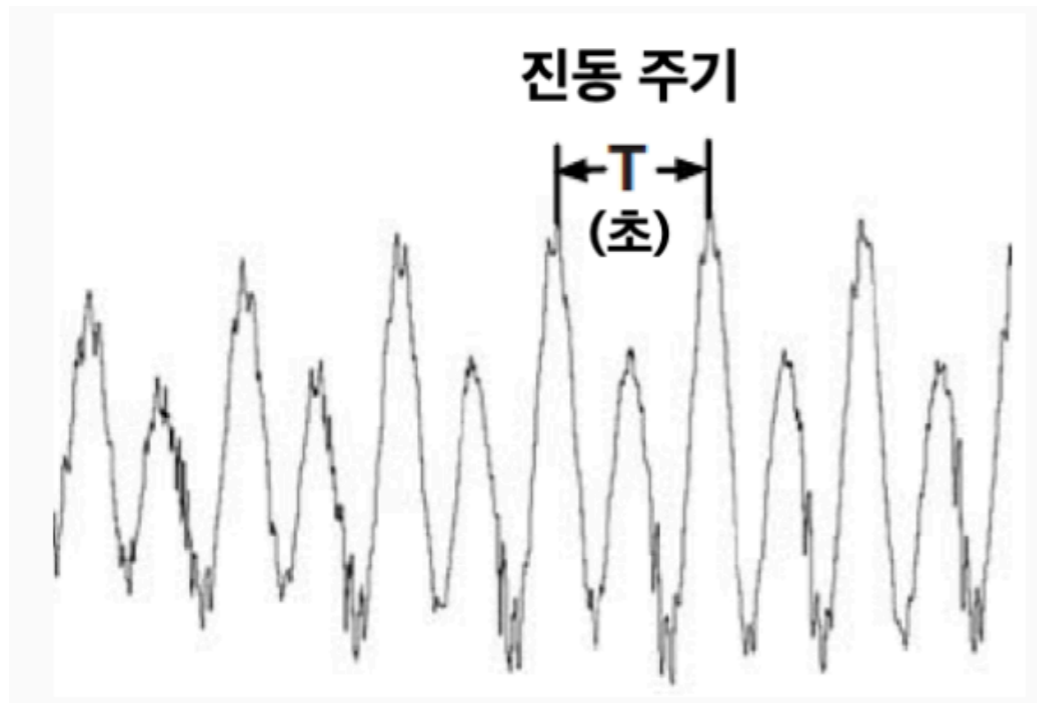
- 음파는 공기를 통하여 사방으로 퍼져나가는데,
- 도중에 벽과 다른 물체의 표면에 닿으면 반사를 하면서 여러 방향으로 퍼져나가다가,
- 궁극적으로 사람의 귀 또는 마이크(microphone)에 도달하여 소리로 감지하게 된다.

# 소리의 특성

- 물체가 진동(oscillation)하면서 공기에 파형을 만들어 소리가 나는데, 이를 **음파(sound wave)**라고 한다.



- 음파의 속성 = **진폭**(amplitude, gain) + **주파수**(frequency).
  - 음파의 상하 진폭은 소리의 크기(loudness, volume)를 결정
  - 주파수는 소리의 높낮이(고음/저음, pitch)를 나타냄



$$\text{주파수} = 1 / T \text{ (Hz)}$$

Hertz  
헤르츠

$$T = 0.00454545 \text{ (초)}$$

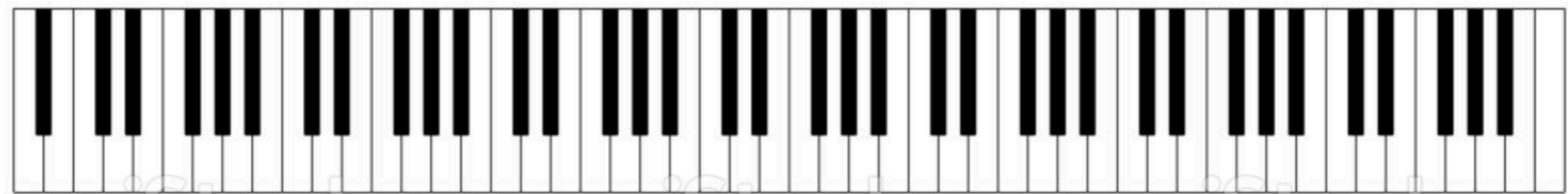
$$\begin{aligned} \text{주파수} &= 1 / 0.00454545 \\ &= 220.0 \text{ (Hz)} \end{aligned}$$

# 소리의 특성

- 물체가 진동(oscillation)하면서 공기에 파형을 만들어 소리가 나는데, 이를 **음파(sound wave)**라고 한다.



- 음파의 속성 = **진폭**(amplitude, gain) + **주파수**(frequency).
  - 음파의 상하 진폭은 소리의 크기(loudness, volume)를 결정
  - 주파수는 소리의 높낮이(고음/저음, pitch)를 나타냄



저음

고음

주파수 작다

주파수 크다

**그럼,  
컴퓨터로 소리를 어떻게 낼까?**

1. 소리의 **합성** = 음파 정보(진폭+주파수)를 디지털로 생성
2. **DAC(Digital-Analog Converter)**로 아날로그 음파로 변환
3. 스피커로 음파 발생

# Chuck

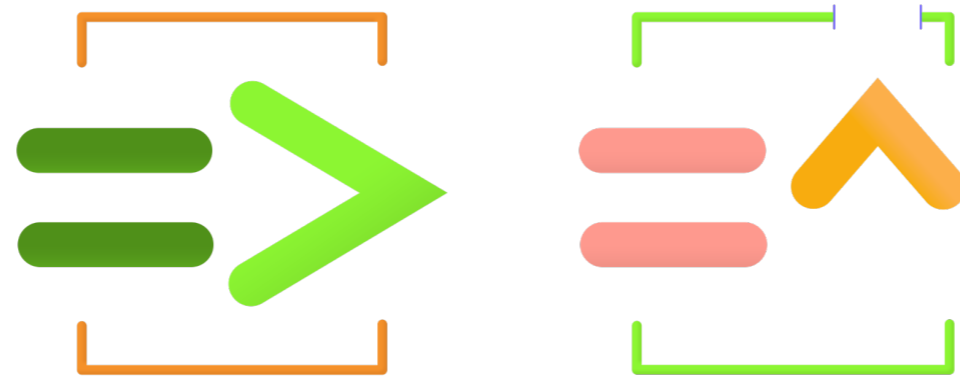
Music Programming Language

strongly-timed | concurrent | on-the-fly

<https://chuck.cs.princeton.edu/>



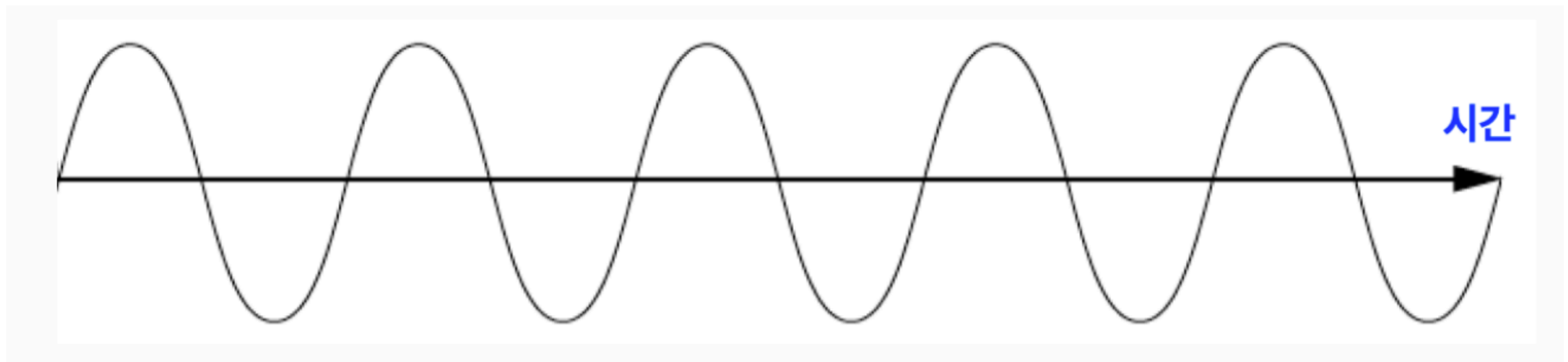
Perry R. Cook



Ge Wang

실시간으로  
소리를 합성하고 음악을 연주하는  
프로그램을 작성할 수 있는  
범용 프로그래밍 언어

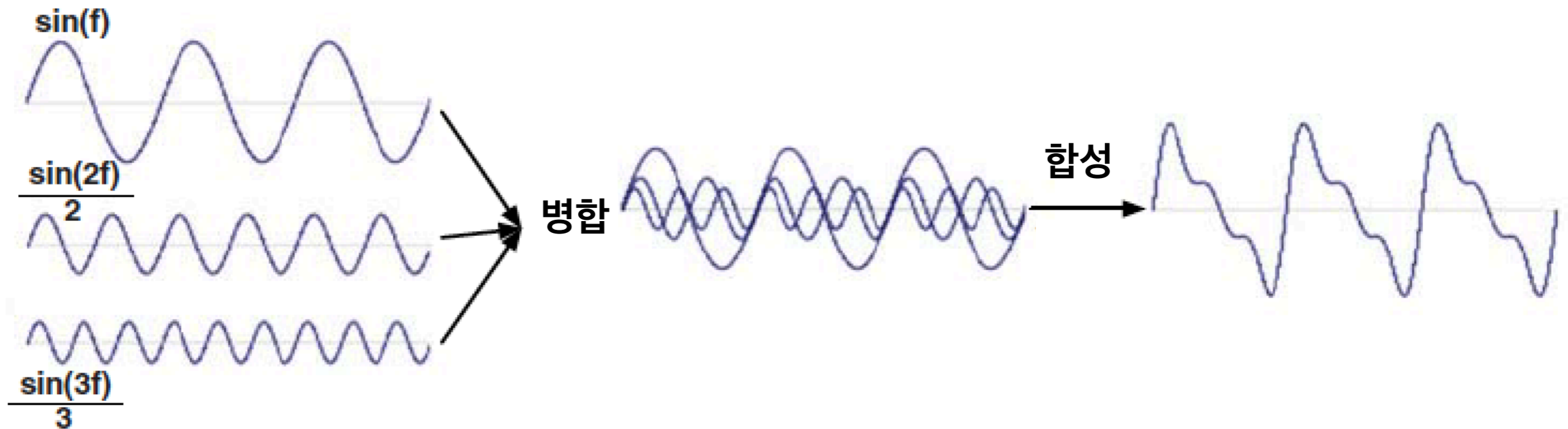
사인파  
Sine wave





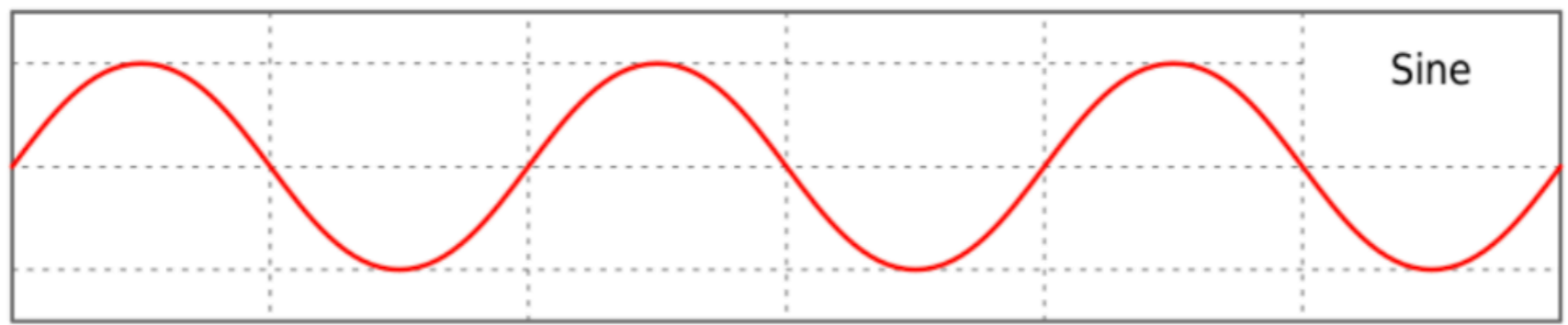
# 소리 합성

다음 3가지를 적절히 조절한 여러 개의 사인파를 병합하여  
주파수 (소리높낮이)  
지연 (delays)  
진폭 (소리크기)  
다른 다양한 모양의 규칙적인 음파를 만들어낼 수 있다.

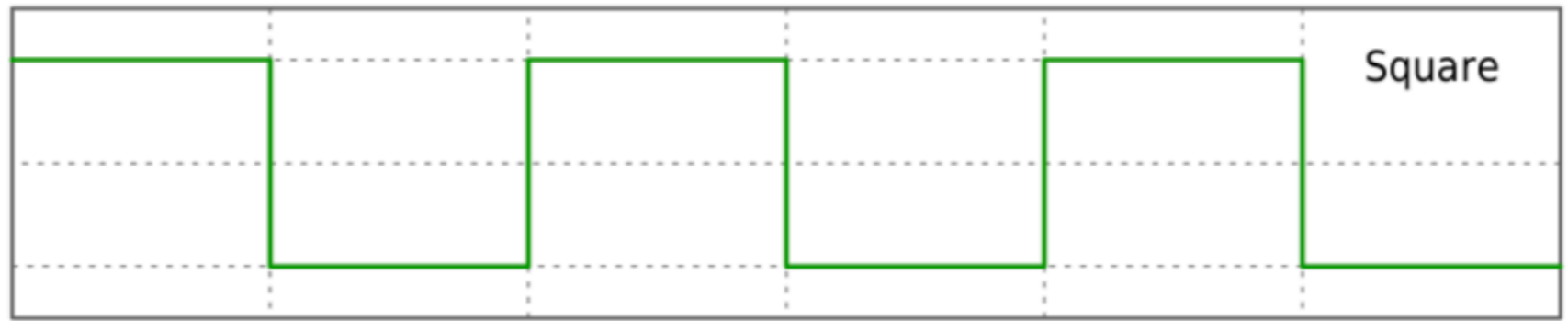


# 내장 진동기 Built-in Oscillators

**SinOsc**



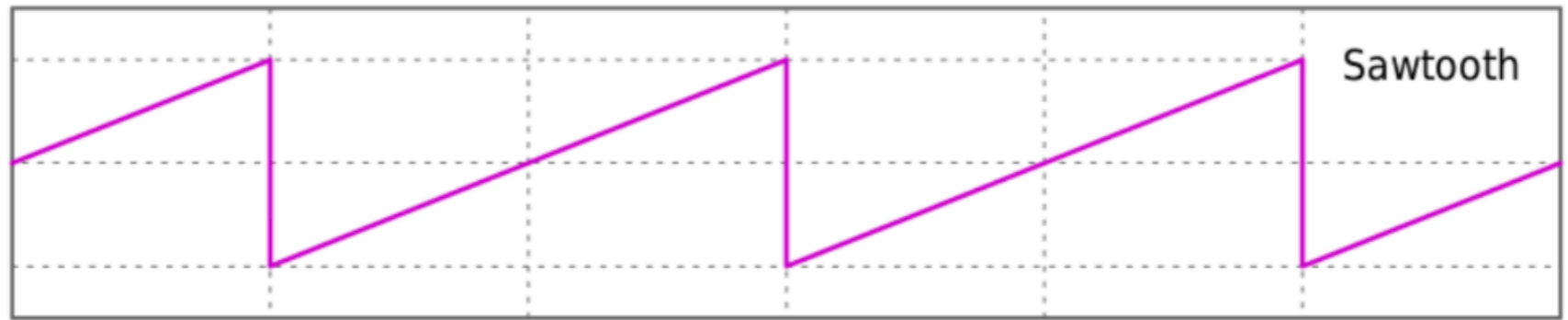
**SqrOsc**



**TriOsc**



**SawOsc**



**clarinet**  
목관악기

**violin**  
현악기

# Chuck

설치



miniAudicle

version 1.5.1.6 (latte)

git: 83d4eee

Copyright (c) Spencer Salazar

Chuck: version 1.5.1.6 (chai) 64-bit  
Copyright (c) Ge Wang and Perry Cook  
<https://chuck.stanford.edu/>



miniAudicle로  
프로그램 실행시켜  
소리 내보기

사인파 진동기를 하나 만들어  
이름을 s 라 하고

디지털-아날로그 변환기에 연결

```
SinOsc s => dac;  
3::second => now;
```

지금부터 3초간 소리를 내라

## 주파수 및 소리크기 변경

```
SinOsc s => dac;  
<<< s.freq(), s.gain() >>>;  
1::second => now;
```

```
440.0 => s.freq;  
<<< s.freq(), s.gain() >>>;  
1::second => now;
```

```
880.0 => s.freq;  
0.5 => s.gain;  
<<< s.freq(), s.gain() >>>;  
1::second => now;
```

s의 주파수 변경

s의 주파수 변경

s의 소리 크기 변경

# 가청 주파수 별 소리 들어보기

s의 소리 크기를 아주 작게

```
SinOsc s => dac;  
0.1 => s.gain;  
while (true) {  
  Math.random2(20, 20000) => s.freq;  
  Math.random2f(30, 600)::ms => now;  
}
```

무한 반복

두 수의 범위 안에서 무작위 수 생성

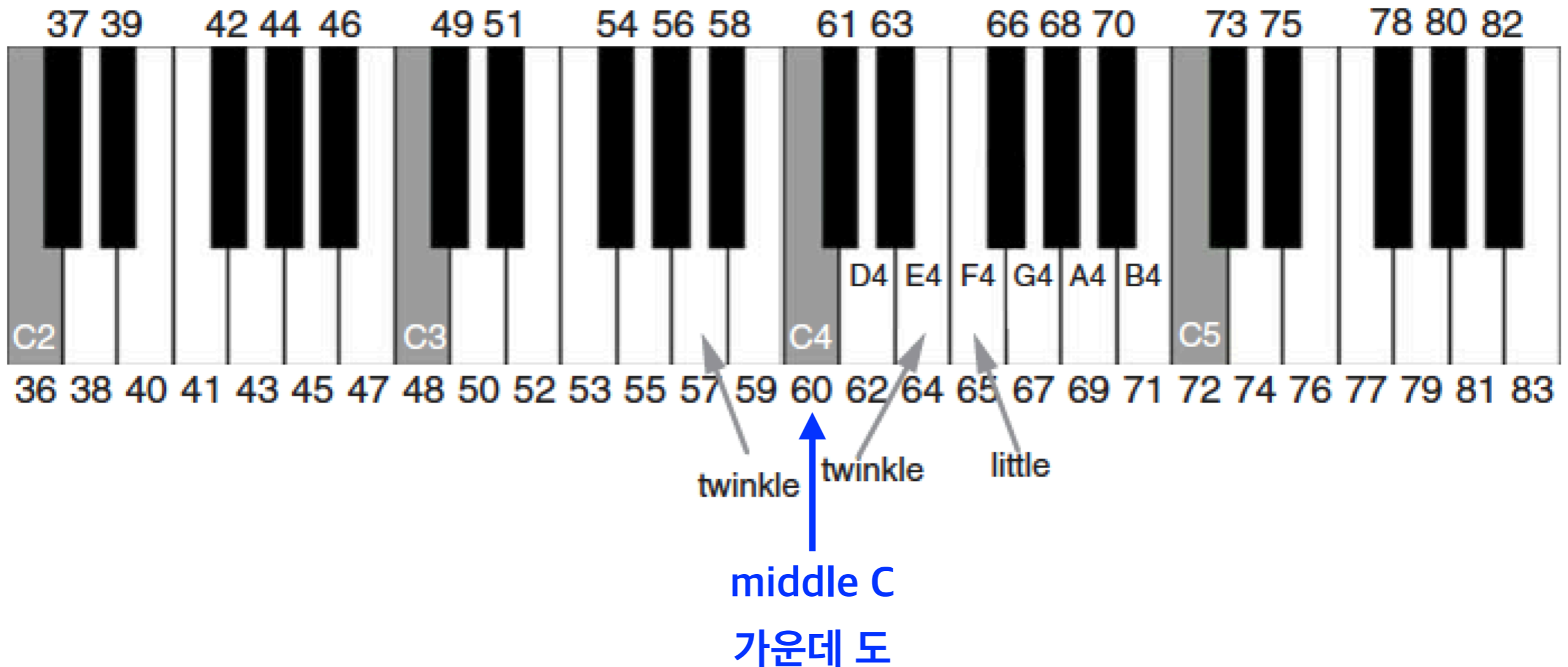
ms = 1/1000 second

가청 주파수 = 20 ~ 20,000 Hz

# MIDI 음 번호

Musical Instrument Digital Interface

0~127





# MIDI 음 번호표

도  
레  
미  
파  
솔  
라  
시

Note	-1	0	1	2	3	4	5	6	7	8	9
C	0	12	24	36	48	60	72	84	96	108	120
C#	1	13	25	37	49	61	73	85	97	109	121
D	2	14	26	38	50	62	74	86	98	110	122
D#	3	15	27	39	51	63	75	87	99	111	123
E	4	16	28	40	52	64	76	88	100	112	124
F	5	17	29	41	53	65	77	89	101	113	125
F#	6	18	30	42	54	66	78	90	102	114	126
G	7	19	31	43	55	67	79	91	103	115	127
G#	8	20	32	44	56	68	80	92	104	116	
A	9	21	33	45	57	69	81	93	105	117	
A#	10	22	34	46	58	70	82	94	106	118	
B	11	23	35	47	59	71	83	95	107	119	

# MIDI 음 들어보기 Random Walk

```
SinOsc s => dac;  
0.5 => s.gain;  
72 => int note;  
while (true) {  
  <<< "MIDI =", note >>>;  
  Std.mtof(note) => s.freq;  
  0.3::second => now;  
  Math.random2(-2, 2) +=> note;  
}
```

s의 소리를 중간크기로

기본 MIDI음 설정

s의 주파수 설정

0.3초간 연주

무한 반복

MIDI 음을 주파수로 변환

MIDI음 -2, -1, 0, +1, +2 무작위 증감

# MIDI 음 들어보기

## 다장조 스케일



다장조 스케일 배열 scale 생성

```
[60, 62, 64, 65, 67, 69, 71, 72] @=> int scale[];  
SinOsc s => dac;  
0.5 => s.gain;  
for (0 => int i; i < scale.size(); i++) {  
  <<< "MIDI =", scale[i] >>>;  
  Std.mtof(scale[i]) => s.freq;  
  0.3::second => now;  
}
```

앞에서부터 차례로 한음 씩 골라

0.3초간 연주

# Stk = Synthesis toolkit (소리합성 툴킷)

## Stk Instrument

Rhodey

BeeThree

Clarinet

PercFlut

Mandolin

...

ModalBar

VoicForm

악기 설정

```
[60, 62, 64, 65, 67, 69, 71, 72] @=> int scale[];  
Rhodey s => dac;  
0.5 => s.gain;  
for (0 => int i; i < scale.size(); i++) {  
  <<< "MIDI =", scale[i] >>>;  
  Std.mtof(scale[i]) => s.freq;  
  1 => s.noteOn;  
  0.3::second => now;  
  1 => s.noteOff;  
}
```

켜기

끄기

# 멜로디 + 반주

동시 계산  
concurrency



1. 푸 른 하 늘 은 - 하 수  
2. 은 하 수 를 건 - 너 서

spork ~

shred 생성

# 돌림노래

동시 계산  
concurrency

## Row Row Row Your Boat

Folk song

1 2  
Row, row, row your boat Gent - ly down the stream.

3 4  
Mer - ri - ly, mer - ri - ly, mer - ri - ly, mer - ri - ly, life is but a dream.

bethsnotes.com

spork ~

shred 생성

## 소리 샘플

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
샘플	0	48	187	204	356	378	365	392	367	298	276	223	186	143	87	25

## CD 음원

샘플 비율 = 초당 44,000개  
샘플 하나 저장 공간 = 16 비트


## 소리 샘플 담는 장치


SndBuf




# 드럼 연주 데모

▼  audio

 hihat\_01.wav

 kick\_01.wav

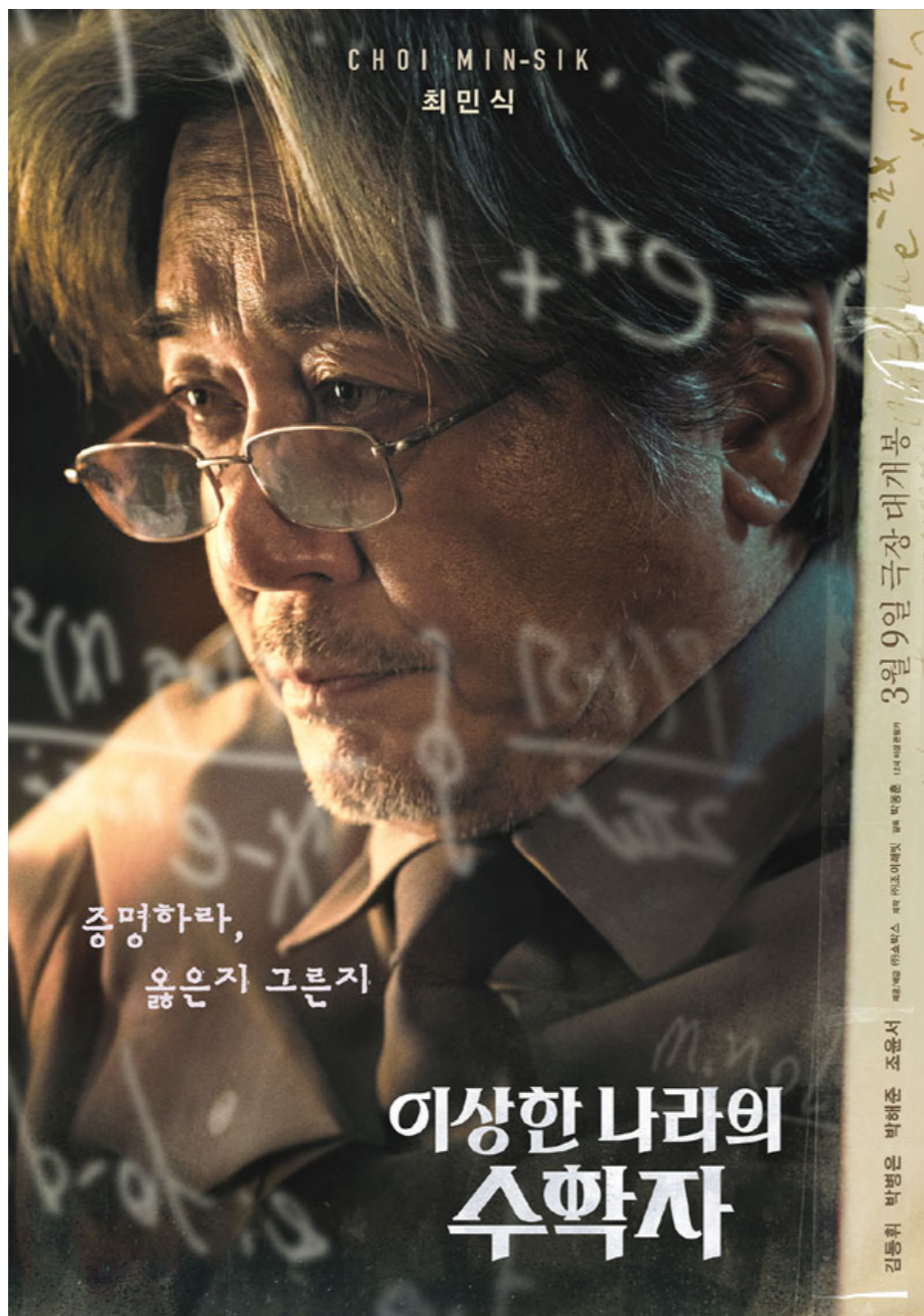
 snare\_01.wav





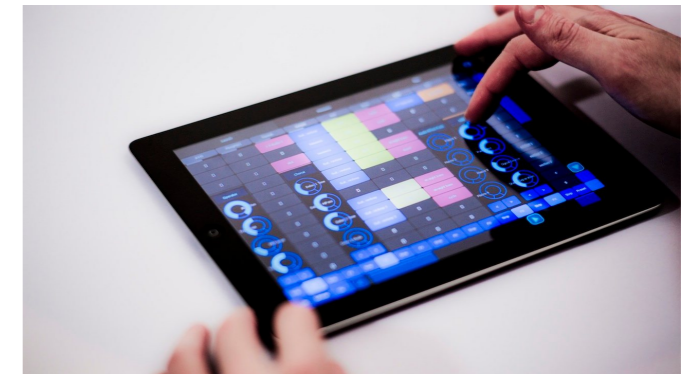
합주

## The Pi Song



# 컴퓨터를 악기로 ~

외부 기기 활용 실시간 비동기 입력에 반응하는  
이벤트 처리 기능 활용



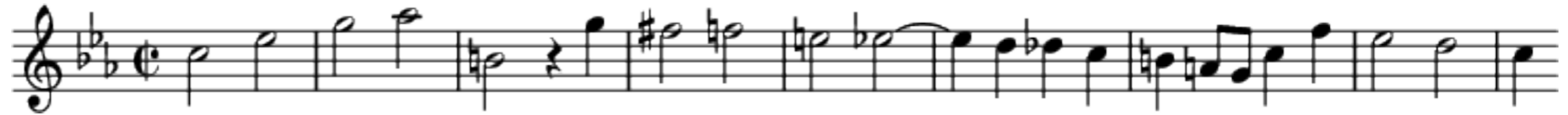
데모

콩이랑 듀엣 (신민경)  
3:18

# Algorithmic Music Composition?



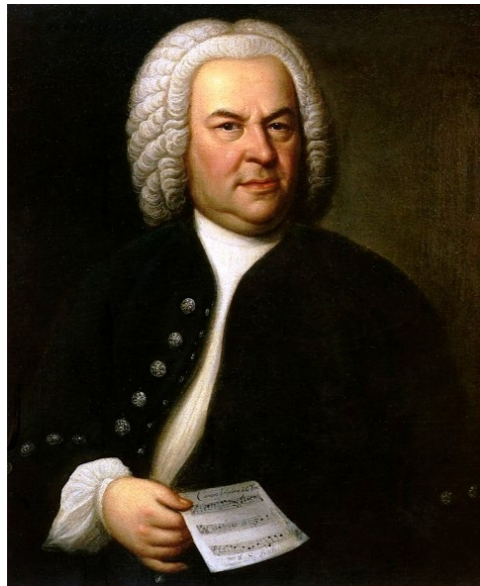
## King Frederick The Royal Theme



## Johann Sebastian Bach

### Musical Offering, BWV 1079

#### Canones diversi super thema regium: Canon a 2 “Crab Canon”



**모두를 위한 SW 지식플러스**

**음악프로그래밍**

**~ 끝 ~**

**Q&A**