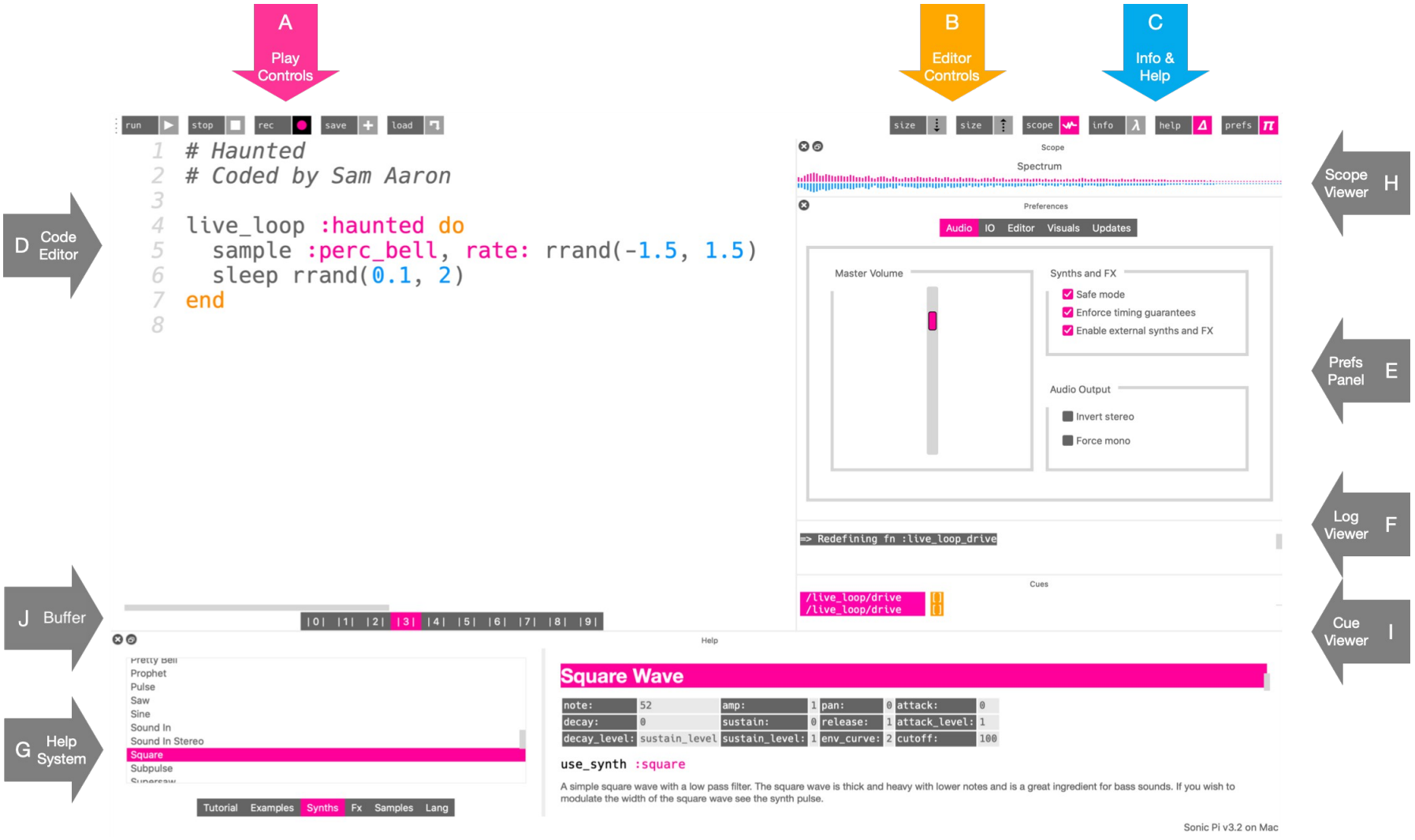
Sonic PI Workshop

Sonic PI


A
Play Controls
B
Editor Controls
C
Info & Help
run
stop
rec
save
load
size
size
scope
info
help
prefs
1 # Haunted
2 # Coded by Sam Aaron
3
4 live_loop :haunted do
5 sample :perc_bell, rate: rrand(-1.5, 1.5)
6 sleep rrand(0.1, 2)
7 end
8
D Code Editor
Scope
Spectrum
Scope Viewer H
Preferences
Audio IO Editor Visuals Updates
Master Volume
Synths and FX
Safe mode
Enforce timing guarantees
Enable external synths and FX
Audio Output
Invert stereo
Force mono
Prefs Panel E
=> Redefining fn :live_loop_drive
Log Viewer F
Cues
/live_loop/drive
/live_loop/drive
Cue Viewer I
J Buffer
|0| |1| |2| |3| |4| |5| |6| |7| |8| |9|
Help
Prophet
Pulse
Saw
Sine
Sound In
Sound In Stereo
Square
Subpulse
G Help System
Tutorial Examples Synths Fx Samples Lang
Square Wave
note: 52 amp: 1 pan: 0 attack: 0
decay: 0 sustain: 0 release: 1 attack_level: 1
decay_level: sustain_level sustain_level: 1 env_curve: 2 cutoff: 100
use_synth :square
A simple square wave with a low pass filter. The square wave is thick and heavy with lower notes and is a great ingredient for bass sounds. If you wish to modulate the width of the square wave see the synth pulse.
Sonic Pi v3.2 on Mac

Sonic PI

- A – Play Controls:
  - "Run" = Ausführen des Codes im Editor.
  - "Stop" = Stoppen des gesamten ausgeführten Codes.
  - "Save" = Speichern des Codes als externen Datei.
  - "Record", um eine Aufzeichnung (eine WAV-Datei) des abgespielten Sounds zu erstellen.
- B – Editor Controls: Schaltflächen kann der Code-Editor bearbeiten.
- C – Info und Hilfe
- D – Code Editor
- E – Prefs Panel: Einstellungen.
- F – Log Anzeige: Bei Code Aufführung wird - werden Informationen zur Funktionsweise des Programms im Log angezeigt.
- G – Hilfe System
- H – Scope Ansicht: Ton sehen, den Sie hören. Die Tonwelle sieht wie eine Säge aus
  - Grundton = Sinuskurve.
  - Größe = Unterschied zwischen lauten und leisen Tönen. Es gibt 3 Bereiche zum Spielen.
- I – Aufrufsansicht: Zeigt an, was gerade aufgerufen wird und was folgt.
- J – Buffer

Sonic PI Buffer 0

- Buffer 0
  - play 60
  - Sleep 1

					Spiele und warte
play 80	60	62	64	65	20
	67	69	71	72	
	:c4	:d4	:c5	:d5	

Sonic PI Buffer 0

- c4 e4 g4
- Spiele einen Akkord (Dreiklang)

---

c5 e5 g5 f4 a4 c5 g4 a4 d5

Sonic PI Buffer 1

- Buffer 1
  - play_chord [:c4, :e4, :g4]
  - sleep 1

Drei Töne gleichzeitig nennt man einen Akkord. So ist es einfacher als vorher. Hier ein C-Akkord.

---

c5 e5 g5 f4 a4 c5 g4 a4 d5

Sonic PI Buffer 2

- Buffer 2
  - Schleifen
    - Ist dafür da, um Melodien oder Töne zu wiederholen
    - z.B. 5 mal, für immer, Zählen von 1 bis 10

Sonic PI Buffer 2

- play_pattern (scale :c4. major)

Spiele ein Muster (=Pattern) – hier eine Tonleiter (scale)

:major :major_pentatonic :minor_pentatonic :minor

Sonic PI Buffer 2

- use_bpm 120
- play_pattern (scale:e4, :minor)

Verwende eine andere Geschwindigkeit b p m = Beat per minute = Schläge pro Minute				
50	240	400	100	600
	:major_pentatonic		:minor_pentatonic	:minor

Sonic PI Buffer 2

- use_bpm 600
- 2.times do
  play_pattern (scale :e4, :minor)
- end

2 times = 2 mal. Wir nennen das eine Schleife

3.times

5.times

Sonic PI Buffer 2

- live_loop :tonleiter do
  use_bpm 120
  play_pattern (scale :e4, :minor)
- end

Wir nennen das Endlosschleife, die während des Spielens aktualisiert werden kann.

Ändere auf 480. drücke RUN und höre, wann die Änderung kommt. Sofort?

Sonic PI Buffer 2

- Füge den folgenden Befehl hinzu

```
use_synth :saw
```

Wie wäre es mit einem anderen Sound unseres Synthesizers?

:dsaw	:mod_dsaw	:prophet	:piano
:blade	:tb303	:pluck	:dtri

Sonic PI Buffer 2

- play_pattern (scale :e4, :minor)
- play_pattern (scale :e4, :minor).reverse

Und nun spielen wir die Tonleiter rückwärts

Sonic PI Buffer 2

- play_pattern (scale :e4, :minor)
- play_pattern (scale :e4, :minor).reverse

Und nun spielen wir die Tonleiter rückwärts

Sonic PI Buffer 2

- play_pattern (scale :e4, :minor)
- play_pattern (scale :e4, :minor).reverse

Und nun spielen wir die Tonleiter rückwärts

Sonic PI Buffer 3

- Buffer 3
  - live_loop :geblubber do
    use_bpm 240
    play_pattern (scale :e4, :minor).choose
    sleep 1
  - end

Choose heißt wählen. Dieser Befehl wählt einen zufälligen Ton aus der Tonleiter. Jedesmal einen anderen

- play spielt nur einen Ton (play_pattern ein Muster, also viele)
- choose wählt einen beliebigen Ton

Sonic PI Buffer 8

- Buffer 8
  - Wenn / Dann
    - Zähle mit if / else

Sonic PI Buffer 8

- meinton = 50
- Play meinton

Wir nennen meinton eine Variable

Was passiert hier und warum?
Was ist meinton? Was bedeutet das =?

Was passiert wenn du meinton = 60 schreibst?

Sonic PI Buffer 8

- meinton = 50
- Play meinton
- sleep 1
- meinton = meinton + 20
- play meinton

Welche Zahl hat meinton nach der vierten Zeile?

Welcher Ton wird in der vierten Zeile gespielt?

Sonic PI Buffer 8

- meinton = 50
- Play meinton
- sleep 1
- meinton = meinton + 20
- play meinton

Welche Zahl hat meinton nach der vierten Zeile?

Welcher Ton wird in der vierten Zeile gespielt?

Sonic PI Buffer 8

```
meinton = 50

play meinton
sleep 1

meinton = meinton +1
play meinton
sleep 1
```

```
meinton = 50

play meinton
sleep 1

meinton = meinton +1
play meinton
sleep 1
```

Welche vier Töne werden hier gespielt?

Sonic PI Buffer 8

- meinton = 50
- live_loop :immerwieder do
    meinton = meinton +1
    play meinton
    sleep 0.5
- end

Was passiert nun?

Sonic PI Buffer 2

```
meinton = 50
live_loop :immerwieder do
    meinton = meinton +1
    play meinton
    sleep 0.5
    if(meinton==100) then
        meinton =50
    end
end
```

Was passiert bei 100 und warum?

Sonic PI Buffer 8

```
meinton = 50
zaehlHoch = 1

live_loop :immerwieder do
  if(zaehlHoch == 1) then
    meinton = meinton + 1
  end
  if(zaehlHoch == 0) then
    meinton = meinton - 1
  end
  play meinton
  sleep 1

  if(meinton == 55) then
    zaehlHoch = 0
  end
  if(meinton == 50) then
    zaehlHoch = 1
  end
end
```

Zum Knobeln für die Fortgeschrittenen

Was passiert hier?

Sonic PI Buffer 4

- Buffer 4
  - live_loop :schlagzeug do
    sample :bp_hous
    sleep 1
  - end

Wir nennen das eine (Endlos-) Schleife

Füge ein weiteres sample sn_zome mit sleep 1 hinzu
Mach' das Schlagzeug schneller (120)

:drum_bass_hard :drum_snare_hard :drum_tom_hi_hard

Sonic PI Buffer 5

- Buffer 5
  - live_loop :melodie do
    sample :guit_em9
    sleep 2
  - end

Ein Gitarren-Beispiel

Probiere aus
Danach kopiere das Schlagzeug (Buffer 4) und Melodie in Buffer 5 zusammen

Sonic PI Buffer 6

- Buffer 6
  - Jetzt fügen wir in Buffer 6 alles zusammen
  - Erst Buffer 5, dann Buffer 3 und Buffer 2
  - Starte nach jedem weiteren Buffer neu
  - Kopiere Buffer 1 und füge eine live_loop hinzu

Benutze auch Size- und Size+, um die Text Größe anzupassen

use_ bpm
use_synth :hoover
use_synth :hollow
,amp:5

Sonic PI Buffer 6

- Weitere Ideen
  - Mehr Schlagzeug
  - Ein Musik-Programm
  - Effekte
  - Samples „loop_“ mit sample_duration()

Sonic PI Buffer 7

- Spiele etwas mit dem Schlagzeug herum

```
use_bpm 120

live_loop :schlagzeug do

  sample :drum_cymbal_closed, amp:    4, rate: 1
  sleep 1

   sample :drum_cymbal_closed, amp: 2

   sample :drum_bass_soft, amp: 4
   sleep 1

   sample :drum_cymbal_closed, amp: 4
   sleep 1

   sample :drum_cymbal_closed, amp: 4

    sample :drum_snare_hard, amp: 4, rate: 1.2
    sleep 1
end
```

Experimentiere

1) Ändere die Lautstärke

2) Ändere die Rate: rate:2 oder rate:-1

3) Ersetze das letzte Sample durch

```
with_fx :reverb, room: 0.5 do
   sample :drum_snare_hard, amp: 4, rate: 1.2
end
```

4) Verändere die Geschwindigkeit

Sonic PI Buffer 7

- Was geht hier vor
- IF = falls / else = andernfalls

Sonic PI Buffer 7

- Effekte

```
live_loop :mitHall do
    with_fx :reverb, room: 0.9 do
    play_pattern (scale :e4, :minor) end
end
```

Experimentiere

- Fx steht für „Effects“ = Effekte. Jeder Effekt kann auch „Parameter“ haben: Hier die Größe des Raums für den Hallàroom: 0.9
- Verwende eine andere Geschwindigkeit
- Verwende play und choose (siehe Seite 9b)
- Füge noch ein Effekt hinzu:
  - with_fx :krush do

- Probiere andere Effekte aus (siehe Fx im Spickzettel)

Sonic PI Buffer 7

- Sample Dauer

```
live_loop :endlos do
    sample :loop_amen
    sleep 4
end
```

Das Schlagzeug soll durchgängig spielen aber die Pause ist zu lang. Versuche die richtige Länge herauszufinden.

Probiere sleep sample_duration(:loop_amen) aus! Was passiert und warum?
Füge folgendes hinter dem Sample-Befehl ein: , rate: 2 Was macht die Rate?
Was passiert mit der Pause? Wie müssen wir die Pause korrigieren?
Zum Spaß: Versuche mal eine rate:-1

Sonic PI Buffer 9

- Buffer 9
  - Schleifen
    - Alien-Geräusche für Spiele

Sonic PI Buffer 9

```
for meinton in 50..65
    play meinton
    sleep 1
end
```


Wir nennen das eine Zähl-Schleife Was passiert hier und warum?
Was ist meinton? Was passiert mit meinton in der Schleife
Was macht play mit meinton?
Was passiert, wenn du meinton in deinen Namen änderst?

Sonic PI Buffer 9

```
live_loop :immerwieder do
    for meinton in 50..65
        play meinton
        sleep 1
    end
end
```

Eine Zählschleife in der Endlosschleife
Was passiert nun?

Sonic PI Buffer 9

```
use_bpm 1200
```

2400? 4800? 9600?

Sonic PI Buffer 9

```
for meinton in 55..70
    play meinton
    sleep 1
end
```

Füge diesen Teil noch hinzu. Vergleiche! Wie wird das klingen?
Höre es dir erst LANGSAM an (use_bpm 480)

Sonic PI Buffer 9

```
for meinton in 60..75
...
for meinton in 65..80
...
for meinton in 70..85
...
```

75.. , 80..
Lass es schneller laufen... Wie hört sich das an?

Spickzettel

```
play 60
```

```
sleep 1
```

```
use_bpm 600
```

```
play :c4
```

```
play_chord [:c4, :e4, :g4]
```

```
play_chord chord(:e4, :major)
:major      :major_pentatonic  :minor_pentatonic :minor
```

```
play_pattern (scale :e4, :minor)
play_pattern (scale :e4, :minor).reverse
```

```
play              (scale :e4, :minor).choose
```

```
use_synth :hollow → :saw, :hoover, :piano …
```

```
2.times do
  …
end
```

```
live_loop :meineEndlosschleife do
  …
end
```

```
sample :bd_haus → :guit_em9 …
```

Spickzettel

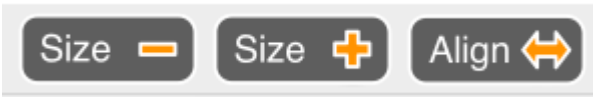
Befehle

ALT-R	Starten	ALT-A	Alles markieren
ALT-S	Stoppen	ALT-C	Kopieren
		ALT-V	Einfügen
STRG-I	Hilfe für Befehl		

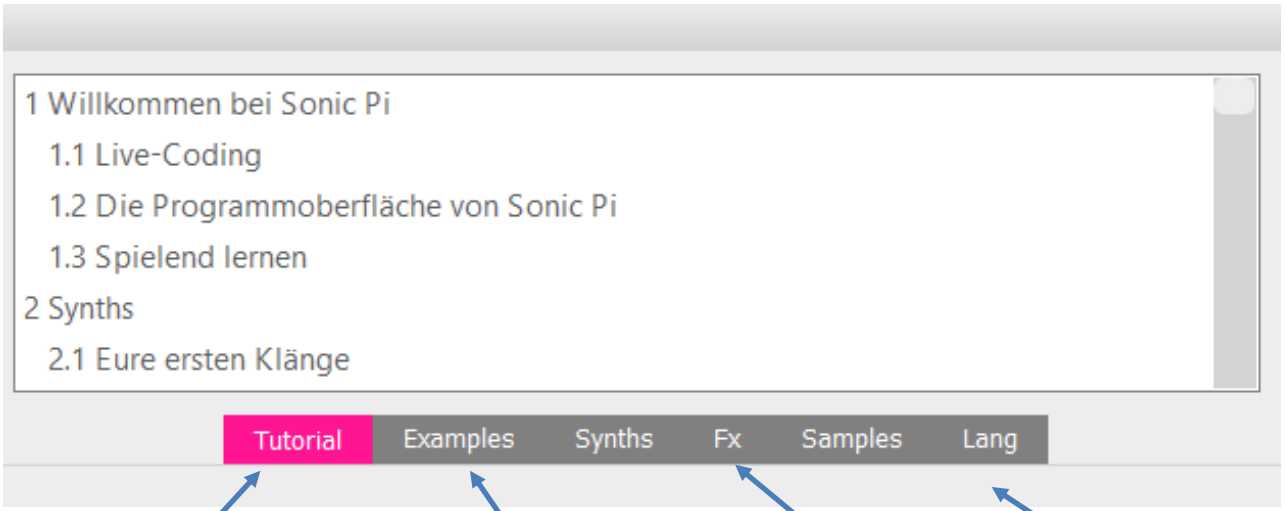
Knöpfe

Starten	Stoppen	Aufnehmen

Speichern	Laden



Text kleiner	Text größer	Text schön machen („ausrichten“)



Spickzettel

use_synth

Tutorial | Examples | **Synths** | Fx | Samples | Lang

:beep	:blade	:bnoise	:cnoise	:dark_ambience		
:dpulse	:dsaw	:dull_bell	:fm	:gnoise	:growl	
:hollow	:hoover					
:mod_beep	:mod_dsaw	:mod_fm	:chiplead	:chipbass	:chipnoise	
:mod_pulse	:mod_saw		:mod_sine	:mod_tri	:pule	
:noise	:piano	:pnoise	:pretty_bell	:prophet	:dtri	:pluck
:saw	:sine	:square	:subpulse	:tb303	:tri	:zawa

sample

Tutorial | Examples | Synths | Fx | **Samples** | Lang

:elec_triangle
:elec_snare
:elec_lo_snare
:elec_hi_snare
:elec_mid_snare
:elec_cymbal
:elec_soft_kick
:elec_filt_snare
:elec_fuzz_tom
:elec_chime
:elec_bong
:elec_twang
:elec_wood
:elec_pop
:elec_beep
:elec_blip
:elec_blip2
:elec_ping
:elec_bell
:elec_flip
:elec_tick
:elec_hollow_kick
:elec_twip
:elec_plip
:elec_blup

:misc_burp
:perc_bell
:perc_bell2
:perc_snap
:perc_snap2
:perc_door
:perc_impact1
:perc_impact2

:bd_ada
:bd_pure
:bd_808
:bd_zum
:bd_gas
:bd_sone
:bd_haus
:bd_zome
:bd_boom
:bd_klub
:bd_fat
:bd_tek
:bd_mehackit

:bass_hit_c
:bass_hard_c
:bass_thick_c
:bass_drop_c
:bass_woodsy_c
:bass_voxy_c
:bass_voxy_hit_c
:bass_dnb_f

:ambi_soft_buzz
:ambi_swoosh
:ambi_drone
:ambi_glass_hum
:ambi_glass_rub
:ambi_haunted_hum
:ambi_piano
:ambi_lunar_land
:ambi_dark_woosh
:ambi_choir

:ambi_soft_buzz
:ambi_swoosh
:ambi_drone
:ambi_glass_hum
:ambi_glass_rub
:ambi_haunted_hum
:ambi_piano
:ambi_lunar_land
:ambi_dark_woosh
:ambi_choir
:ambi_sauna

:drum_heavy_kick
:drum_tom_mid_soft
:drum_tom_mid_hard
:drum_tom_lo_soft
:drum_tom_lo_hard
:drum_tom_hi_soft
:drum_tom_hi_hard
:drum_splash_soft
:drum_splash_hard
:drum_snare_soft
:drum_snare_hard
:drum_cymbal_soft
:drum_cymbal_hard
:drum_cymbal_open
:drum_cymbal_closed
:drum_cymbal_pedal
:drum_bass_soft
:drum_bass_hard
:sn_dub
:sn_dolf
:sn_zome
:sn_generic

:glitch_bass_g
:glitch_perc1
:glitch_perc2
:glitch_perc3
:glitch_perc4
:glitch_perc5
:glitch_robot1
:glitch_robot2

:loop_industrial
:loop_compus
:loop_amen
:loop_amen_full
:loop_garzul
:loop_mika
:loop_breakbeat
:loop_3d_printer
:loop_drone_g_97
:loop_electric
:loop_mehackit1
:loop_mehackit2

:drum_cowbell
:drum_roll
:misc_cros
:misc_cineboom
:perc_swash
:perc_till
:loop_safari
:loop_tabla
:loop_perc1
:loop_perc2
:loop_weirdo

:guit_harmonics
:guit_e_fifths
:guit_e_slide
:guit_em9