

Livecoding Workshop from Sketch to Live

2016.12.9

Art & Science gallery lab AXIOM

田所淳

このワークショップについて

- ▶ ライブコーディング(Livecoding)による表現を探求するワークショップシリーズ
- ▶ 今回はイントロダクション
- ▶ 2017年より、シリーズ本格開始予定!



このワークショップについて

- ▶ ワークショップの担当：田所 淳 (クリエイティブコーダー)
- ▶ フリーランスでのプログラマーの他、大学で非常勤講師など
- ▶ 最近はライブコーディングの研究中

このワークショップについて

▶ <http://yoppa.org/>

yoppa.org

GEIDAI

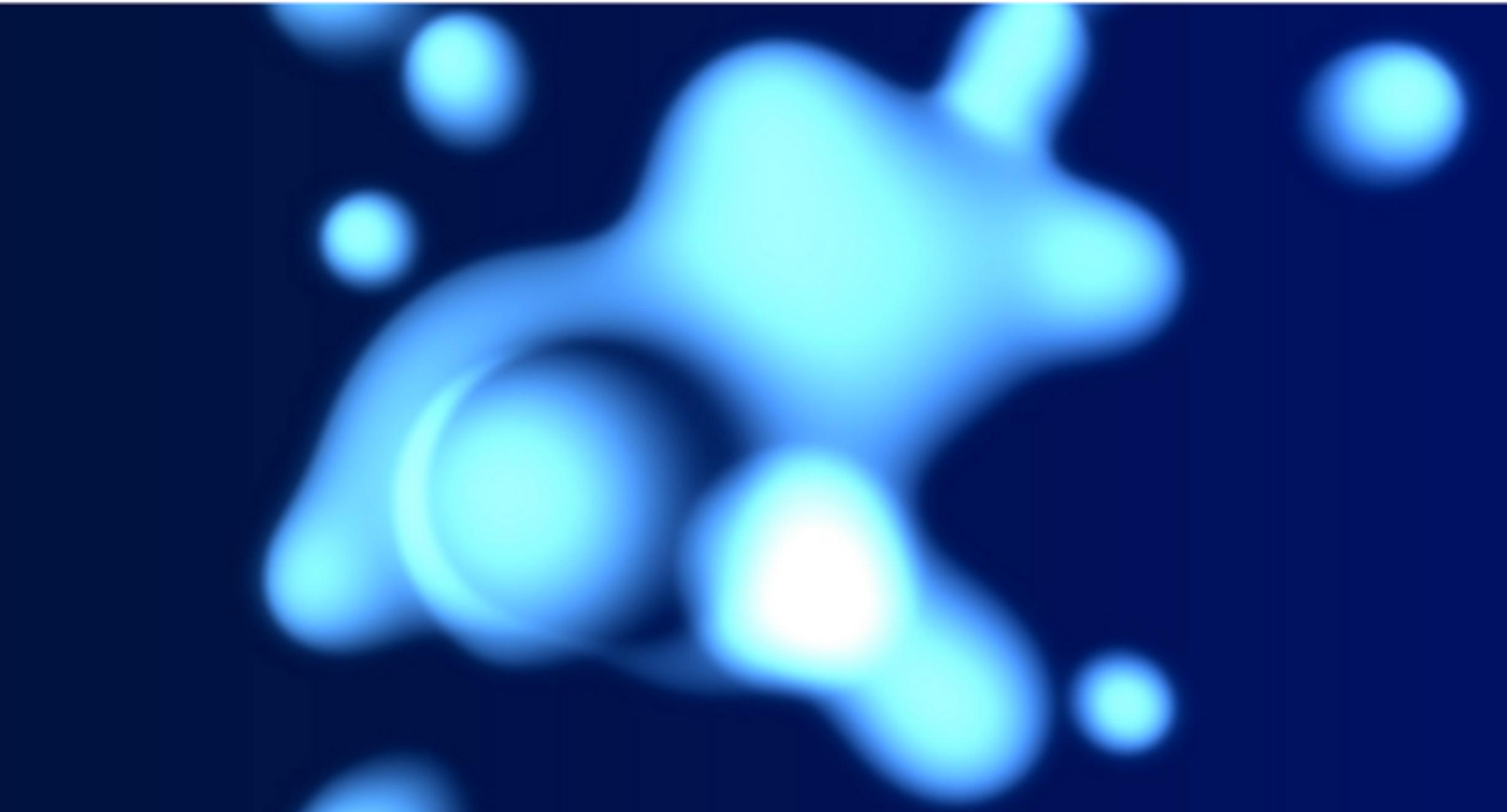
- infomation edit
- art media design

TAMABI

- mediaart basic
- bmaw 13
- iTamabi 13

LINK

- yoppa blog
- openFrameworks.jp
- artsat.jp



April 6, 2012

About this site

このサイトは田所淳の講義、仕事、日記、そのほかにしました。基本的にリンクはフリーです。どの自由に利用していただいて構いません。記述の誤りや不適切な表現などございましたら、ご指摘ください。

このワークショップについて

- ▶ 『Beyond Interaction[改訂第2版] -クリエイティブ・コーディングのためのopenFrameworks実践ガイド』絶賛販売中!!



Ice Break: ヒューマンライブコーディング

Sketch → Live

Sketch → Live

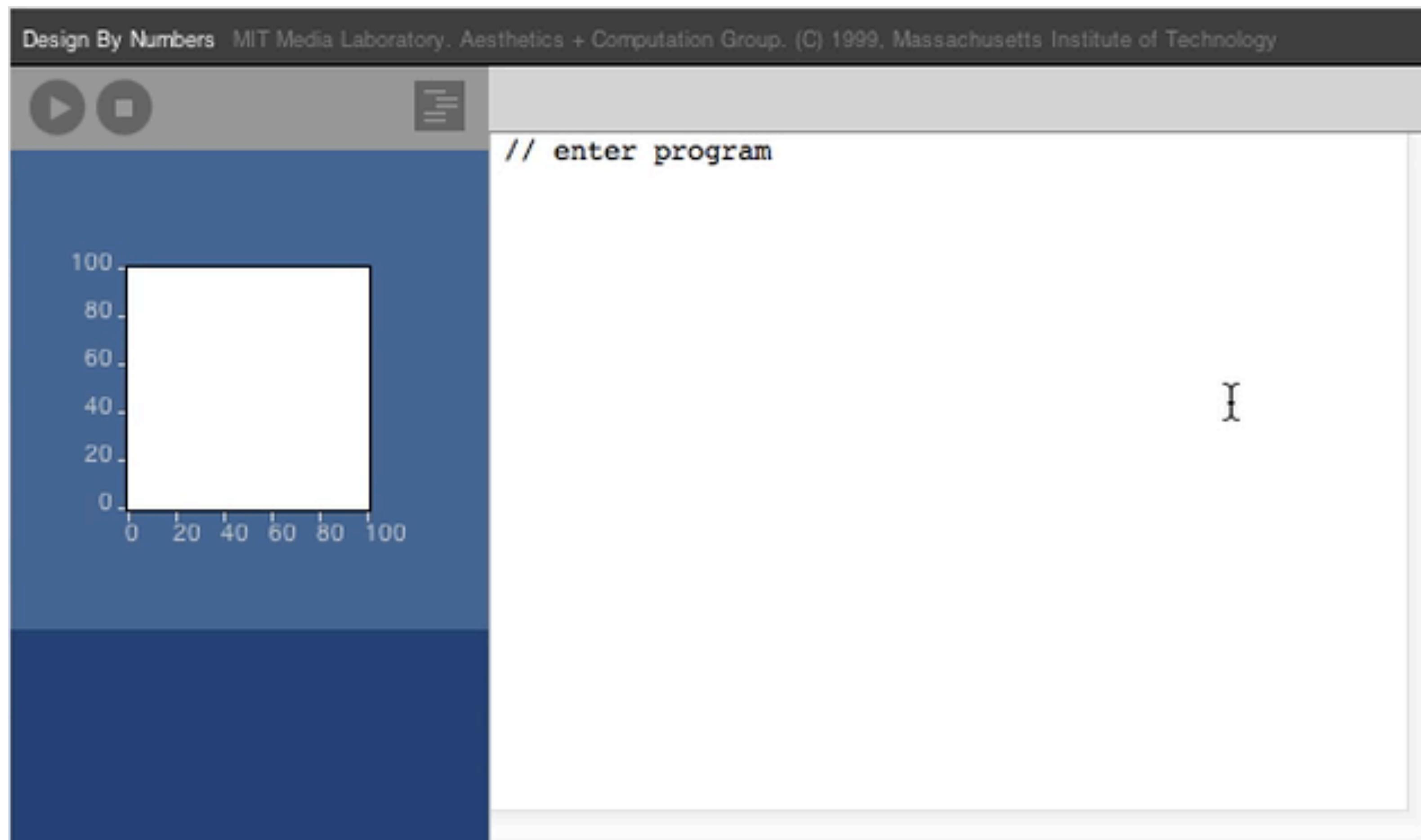
- ▶ 「スケッチ」から「ライブ」へ
- ▶ このワークショップのテーマ
- ▶ その意図を解説

Sketch → Live

- ▶ Sketch
 - ▶ Design by Numbers (1999)
 - ▶ Processing (2001)
 - ▶ openFrameworks (2005)
-
- ▶ スケッチするようにプログラミング
 - ▶ クリエイティブコーディングのムーブメントへ

Sketch → Live

- ▶ Design by Numbersのデモ動画 <https://vimeo.com/72611093>
- ▶ paper, pen など、スケッチをメタファーにしている



Sketch → Live

- ▶ もっと自由にコーディングできないのか?
- ▶ コーディング → 実行 → デバッグ → 実行 → デバッグ …
- ▶ この輪から抜け出したい!

Sketch → Live

- ▶ Bret Victor - Inventing on Principle
- ▶ 2:30秒あたりから始まるデモに注目 <https://vimeo.com/36579366#t=150s>

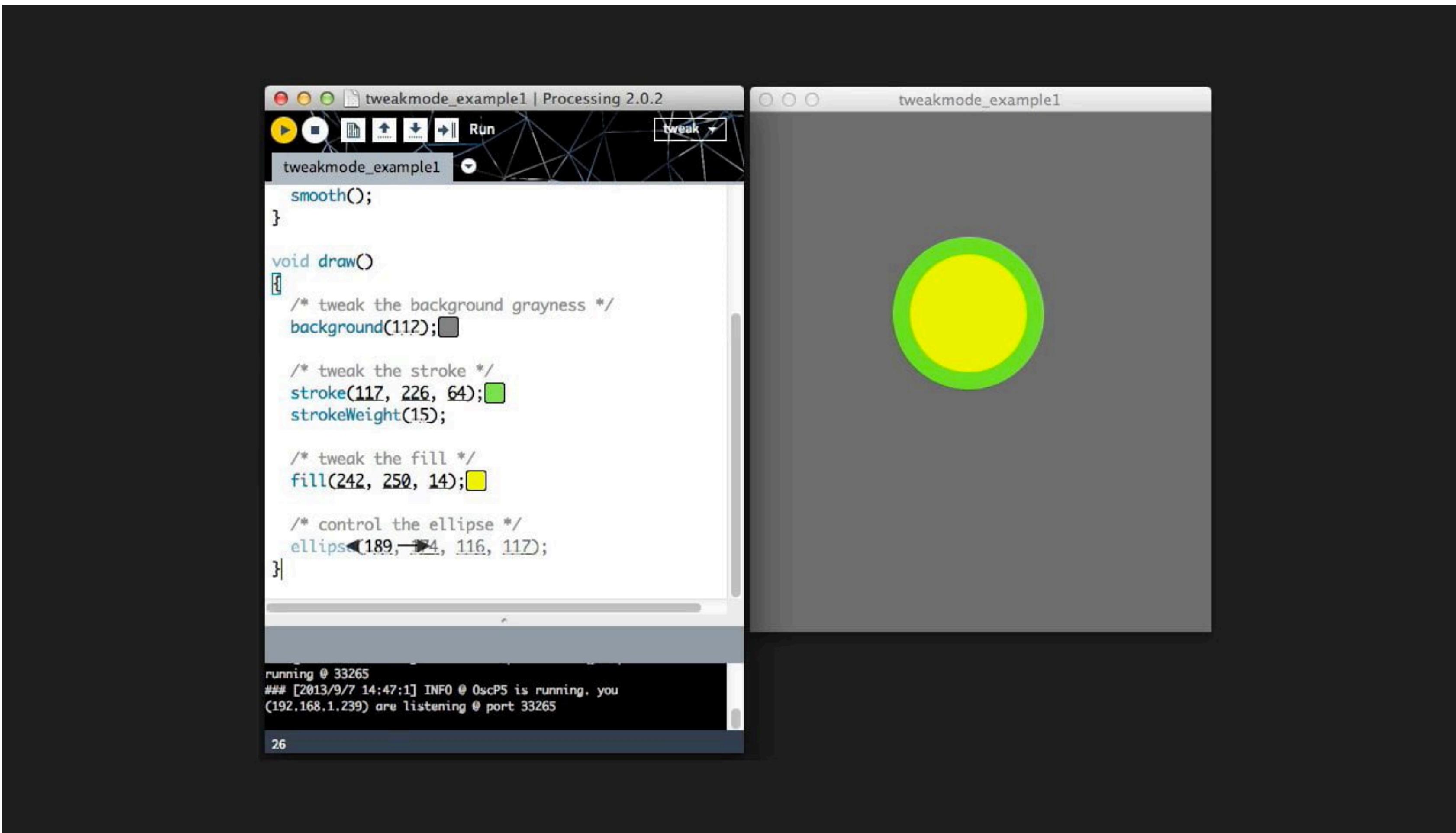


Sketch → Live

- ▶ 実行させたままの状態で、コードに変更を加えていく
- ▶ 変更結果がすぐにフィードバックされる
- ▶ これがライブコーディング!!

Sketch → Live

- ▶ TwitterのTweakモード



Sketch → Live

▶ Swift Playgrounds

The screenshot shows the Swift Playgrounds interface with three main sections:

- Top Left:** A code editor window titled "Balloons.playground - Edited". It contains Swift code for a game scene setup. Key parts include:
 - A function `func doDidMoveToView(scene : SKScene, delegate : SKPhysicsContactDelegate) {`.
 - Comments for "Blimp Control" and "Scene Configuration".
 - Code to set up balloon lighting and per-pixel collisions.
 - Code to load images for balloon explosion.
 - A turbulent field force setup involving `SKFieldNode`, `noiseFieldWithSmoothness(0.7, animationSpeed:0.8)`, and `turbulence.categoryBitMask = WIND_FIELD_CATEGORY`.
 - Initialization code for a hero, fan, and cannons.
 - A contact handling function `func handleContact(bodyA : SKSpriteNode, bodyB : SKSpriteNode) {` that removes balloons from the hero's body.
- Top Right:** A preview window titled "Balloons" showing a 3D game scene. It features a blue sky, a green ground, a red blimp, several colorful balloons (red, green, blue, yellow), a Ferris wheel, and two red cannons on the left.
- Bottom Right:** A graph window titled "let y = 80 * sin(x)" showing a periodic sine wave plotted against time. The x-axis is labeled "- 30 SEC +".

Sketch → Live

- ▶ 音楽のジャンルでは、もっと早い時期からライブコーディングの試みがされていた
- ▶ ライヴエレクトロニクス (Live electronic music) から派生したジャンルとして

Sketch → Live

- ▶ 1990's Laptronica



Sketch → Live

- ▶ 高度に発達した、Laptronica → 何やってるか、わからない
 - ▶ 渾身の力を込めて、フェーダー操作 (??)
 - ▶ ひょっとして、CDをかけてると変わらない (???)
-
- ▶ → 「画面を晒せ (Show Us Your Screens)」

Sketch → Live

- ▶ コードで演奏する様子を常に見せながら(Show your screen)パフォーマンス
- ▶ その場で生成される音のみでパフォーマンス

Sketch → Live

- ▶ 参考: TOPLAP Manufest <http://toplap.org/wiki/ManifestoDraft>
- ▶ Give us access to the performer's mind, to the whole human instrument.
- ▶ Obscurantism is dangerous. **Show us your screens.**
- ▶ Programs are instruments that can change themselves
- ▶ The program is to be transcended - Artificial language is the way.
- ▶ Code should be seen as well as heard, underlying algorithms viewed as well as their visual outcome.
- ▶ Live coding is not about tools. Algorithms are thoughts. Chainsaws are tools. That's why algorithms are sometimes harder to notice than chainsaws.

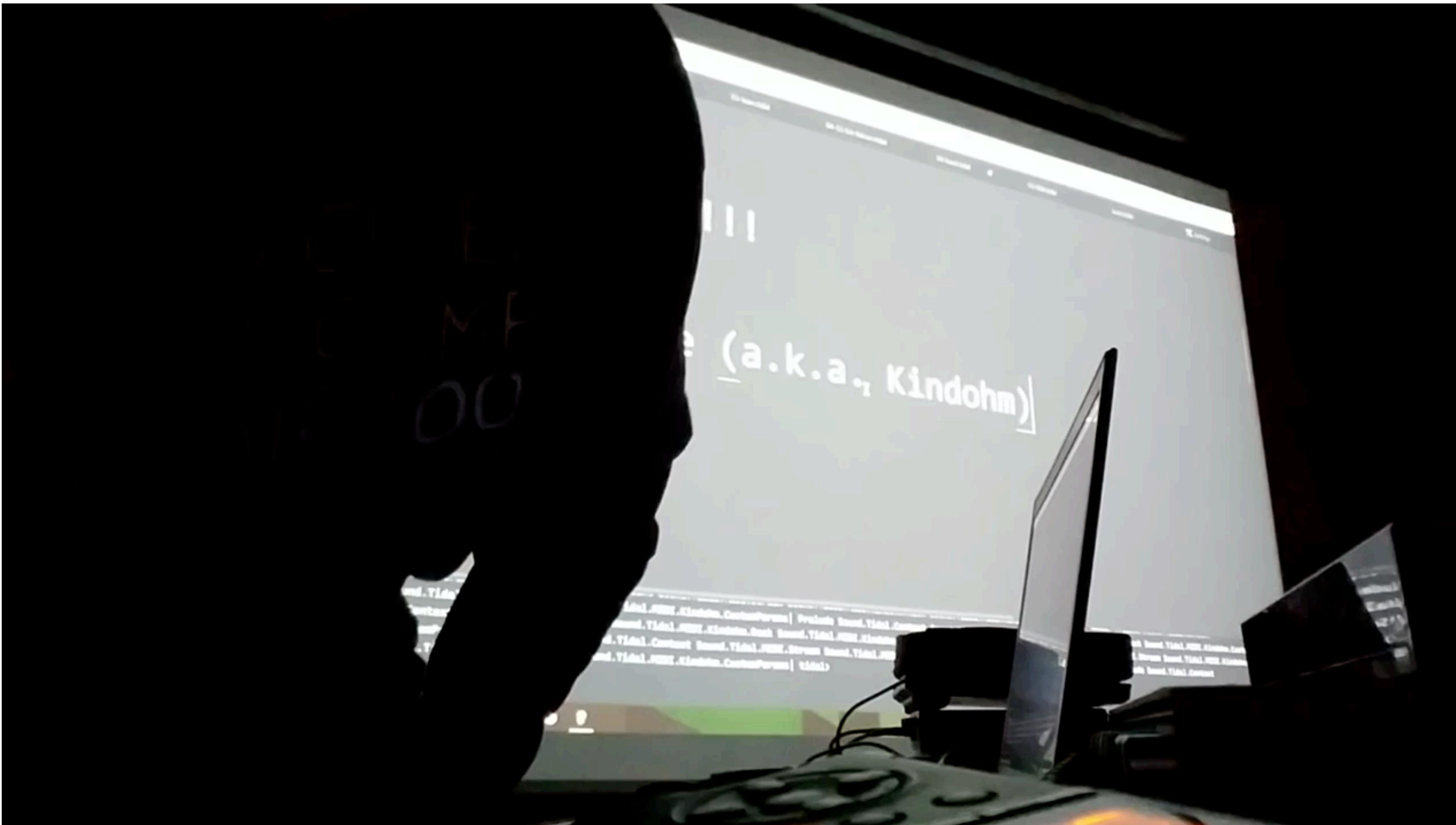
Sketch → Live

- ▶ Show Us Your Screens <https://vimeo.com/20241649>

~doc

Sketch → Live

- ▶ Kindohm Live @ ICLC 2016, Hamilton, Ontario
- ▶ <https://www.youtube.com/watch?v=smQOjFt8e4Q>



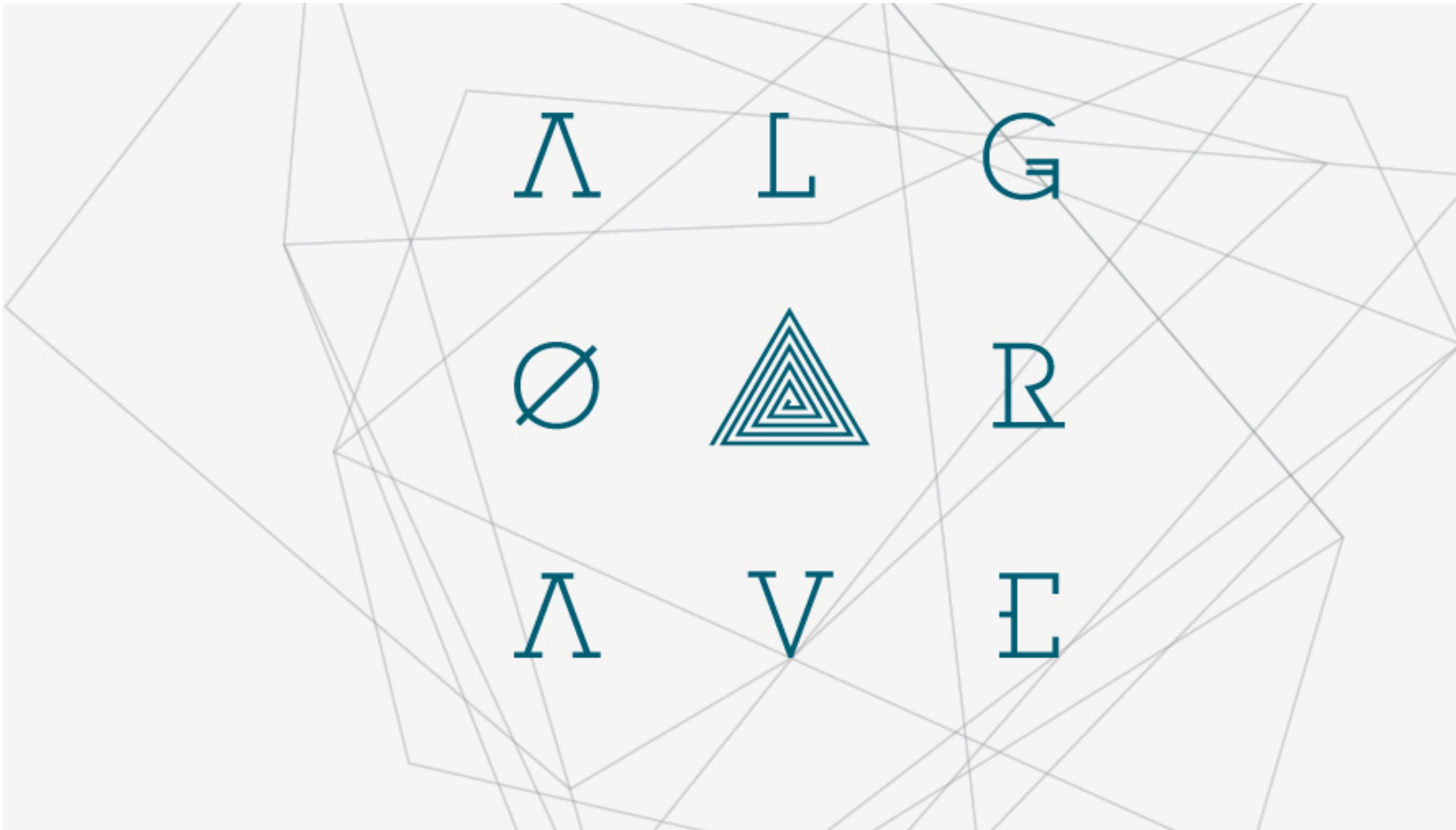
Sketch → Live

- ▶ ICLC (International Conference on Live Coding)
- ▶ <http://iclc.livecodenetwork.org/>



Sketch → Live

- ▶ Algorave <http://algorave.com/>



Sketch → Live

- ▶ TOPLAP <http://toplap.org/>

TOPLAP
...•••

About Live Coding and TOPLAP
TOPLAP wiki (manifesto, software, etc)
Local TOPLAP nodes
Chat (slack)
Discussion forum
Suggest a link / contact
Credits

Events
Call for submissions
Pictures
Music
Videos
Software

TOPLAP Tweets
Flickr
Facebook
Other tweets about livecoding

WOOD STREET GALLERIES
50 Wood Street, Galerie Reichenbach 12-16, Empfangshalle 90-10, 80755 Munich, Germany

Uncategorized

ChoreoGraphic Coding

Here follows a guest post from Joana Chicau, giving some of the context behind "WebPage Act I,II,III", her performance at the second International Conference on Live Coding. This performance was very well received, and brought together the threads of live

October 24, 2016 / No comments

Events

ICLC is go!

The 2016 edition of International Conference on Live Coding is on, we've already had a day of workshops and evening of performances and well into this morning's first paper session. You can follow along on the #iclc2016 channel on the TOPLAP slack, and/or

October 13, 2016 / No comments

Call for submissions, Events

AlgoMech festival, Sheffield, 12-19th November 2016

AlgoMech – the festival of Algorithmic and Mechanical Movement – is coming to Sheffield from the 12th-19th November. It brings together algorithmic and mechanical approaches in

Events

Troy Algorave

Algorave at the Tech Valley Center

Sources and images: www.troyalgorave.com
by the author of a collection of repetitive conditions

Sketch → Live

- ▶ このワークショップを通して、ライブコーダーを目指しましょう!

Sonic Piで、ライブコーディングを始めよう!

Sonic Piで、ライブコーディングを始めよう!

- ▶ ライブコーディングのための開発環境
- ▶ いろいろなアプリケーション、言語が存在している
 - ▶ SuperCollider (JIT Lib)
 - ▶ Chuck
 - ▶ impromptu
 - ▶ extempore
 - ▶ overtone
 - ▶ Gibber
 - ▶ TidalCycles
 - ▶ …etc
- ▶ 参考: <http://toplaphub.org/wiki/ToplapSystems>

Sonic Piで、ライブコーディングを始めよう!

- ▶ SonicPi : おそらく一番導入が簡単な環境
- ▶ <http://sonic-pi.net/>



Sonic Pi **Audible Computing.**

A free **sound synthesiser** for **live coding** designed to support **computing** and **music** lessons within schools.

Use **code** to compose and **perform** in classical and contemporary styles ranging from **Canons** to **Dubstep**.

*Brought to you by [Sam Aaron](#) and the Sonic Pi Core Team,
v2.3.0 is available free for:*

[Raspberry Pi](#)

[Mac OS X](#)

[Windows](#)

Sonic Piで、ライブコーディングを始めよう!

- ▶ Sonic Pi(ソニックパイ)とは?
- ▶ 学校でのプログラミングや音楽の授業をサポートするように設計された、ライブコーディング可能な無料のシンセサイザー
- ▶ 「カノンからダブステップまで」古典～現代の音楽を作曲できる
- ▶ Raspberry Piはもちろん、Mac OS XやWindowsでも動かせる



Sonic Piで、ライブコーディングを始めよう!

- ▶ まずは、OSにあわせたバージョンをダウンロード



Sonic Pi

The Live Coding Music Synth for Everyone.

Simple enough for computing and music lessons.

Powerful enough for professional musicians.

Free to download with a friendly tutorial.

Learn to code creatively by composing or performing **music** in an incredible range of styles from **classical** to **algorave**.

Please help - Sonic Pi needs funding to continue...

Brought to you by [Sam Aaron](#) and the Sonic Pi Core Team

v2.10.0 is available **free** for:

Raspberry Pi

Mac OS X

Windows

Linux



Sonic Piで、ライブコーディングを始めよう!

▶ 起動してみる - まずは画面の構成を理解する



Sonic Pi、プログラミング入門

Sonic Pi、プログラミング入門

- ▶ まずは、エディターに以下のコードを入力して「RUN」を押してみる

```
play 60
```

Sonic Pi、プログラミング入門

- ▶ 数値を変化させてみる → 何が変化した?

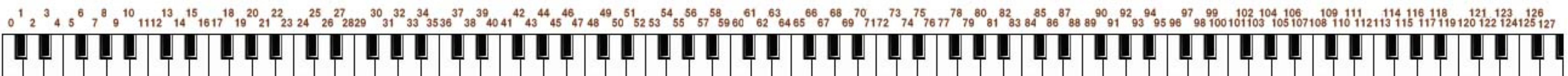
```
play 65
```

Sonic Pi、プログラミング入門

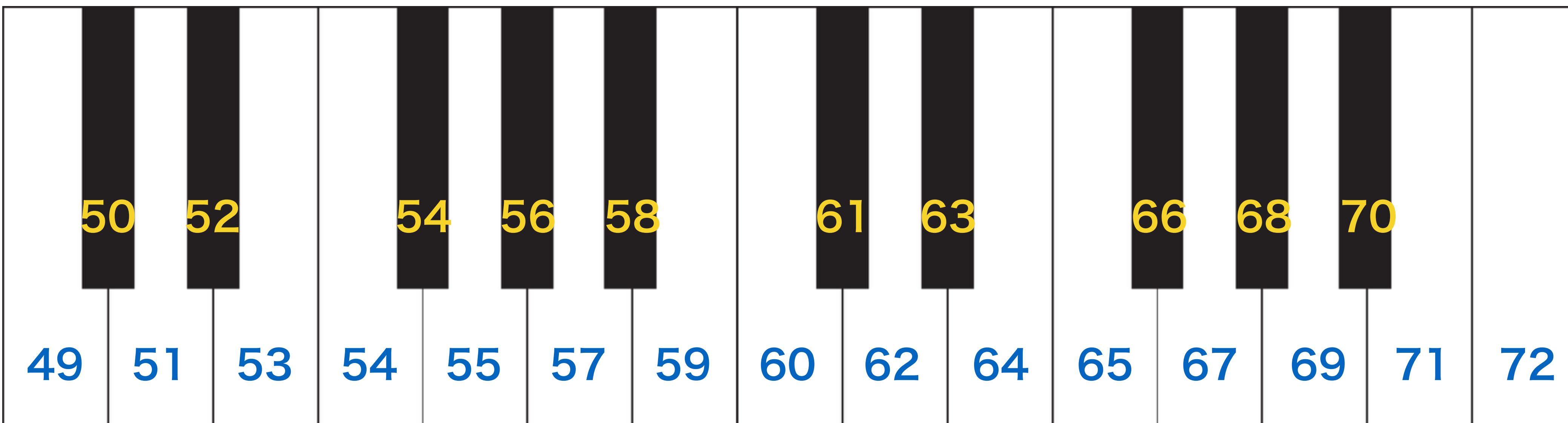
- ▶ 低い数値は低い音を、高い数値は高い音を生み出す
- ▶ では、具体的な数は何を基準にしているのか？

Sonic Pi、プログラミング入門

- ▶ 厳密に言うと：
- ▶ playの後の数字は、鍵盤の番号（ノートナンバー）をあらわしている



- ▶ 中央付近を拡大すると… 60が中心の「ド (C)」



Sonic Pi、プログラミング入門

- ▶ 今度は、複数の「play」を書いてみる → どうなった？

```
play 60
```

```
play 64
```

```
play 67
```

Sonic Pi、プログラミング入門

- ▶ 和音 (Harmony)
- ▶ 複数のplayを書くと、全て同時に演奏される → 和音

- ▶ どの数値がいい組み合わせか？どれがひどい音？
- ▶ 探求しながら、自分自身で見つけてみる

Sonic Pi、プログラミング入門

- ▶ 音と音の間に、sleepを入れてみる

```
play 60
sleep 1
play 64
sleep 1
play 67
```

Sonic Pi、プログラミング入門

- ▶ 一音ずつ、同時ではなく演奏したい場合は？
 - ▶ 音符の間にsleepを入れる
 - ▶ 旋律(メロディー)を奏でる
-
- ▶ sleep 1の1は、これは一拍(1 beat)休む、という意味
 - ▶ 数値を変えると、アルペジオのスピードが変化する

Sonic Pi、プログラミング入門

- ▶ sleepの間隔を、1 → 0.5に

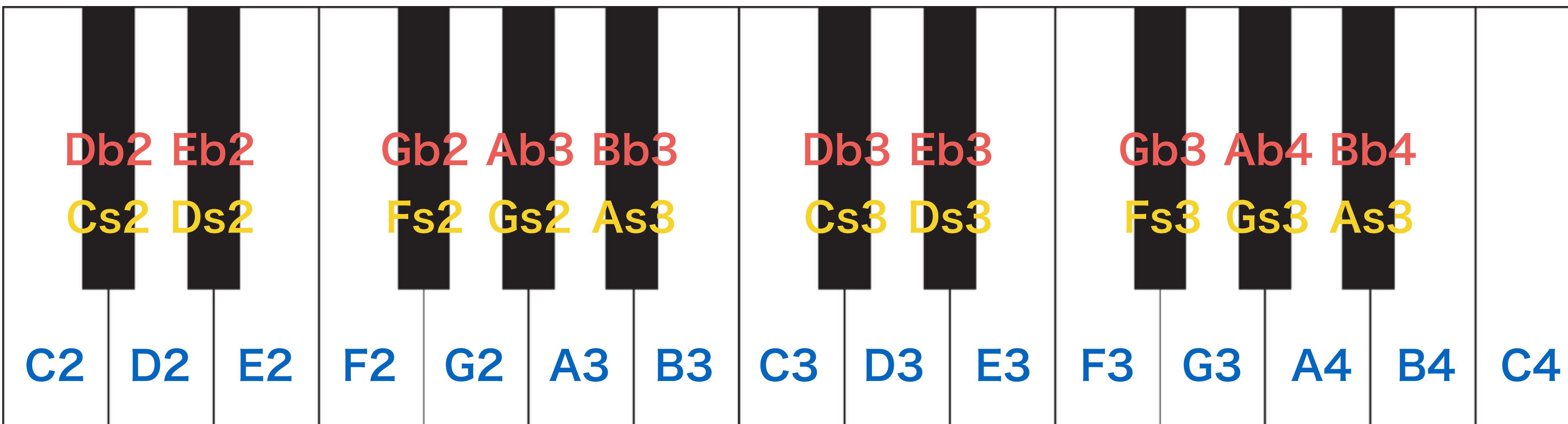
```
play 60
sleep 0.5
play 64
sleep 0.5
play 67
```

Sonic Pi、プログラミング入門

- ▶ 音程は、音階(音の名前)で指定することも可能
 - ▶ : (コロン) の後にアルファベットで指定
-
- ▶ ド → :C
 - ▶ レ → :D
 - ▶ ミ → :E
 - ▶ ファ → :F
 - ▶ ソ → :G
 - ▶ ラ → :A
 - ▶ シ → :B

Sonic Pi、プログラミング入門

- ▶ 音階で表記してみる
- ▶ 音階名とオクターブの数字で表現する
- ▶ #(シャープ 半音上)は「s」 → 例: Cs3
- ▶ b(フラット 半音下)は「b」 → 例: Db3



Sonic Pi、プログラミング入門

- ▶ 音階で指定してみる

```
play :C  
sleep 0.5  
play :E  
sleep 0.5  
play :G
```

Sonic Pi、プログラミング入門

- ▶ 音階 + オクターブ

```
play :C5
sleep 0.5
play :E5
sleep 0.5
play :G5
```

Sonic Pi、プログラミング入門

- ▶ 音階 + オクターブ + #

```
play :C5
sleep 0.5
play :Fs5
sleep 0.5
play :G5
```

Sonic Pi、プログラミング入門

- ▶ シンセのオプション - AmpとPan
- ▶ Amp : 音の大きさ
- ▶ Pan : 音の定位 (左右の位置)

Sonic Pi、プログラミング入門

- ▶ Amp - 「amp: 数値」で指定する
- ▶ 標準の音量が1、そこから相対的な数値で
- ▶ 例: amp:0.5

Sonic Pi、プログラミング入門

- ▶ Pan - 「pan: 数値」で指定する
- ▶ 0が中心 -1が左 1が右
- ▶ 例: pan: -0.75

Sonic Pi、プログラミング入門

- ▶ アンプとパンを指定してみる

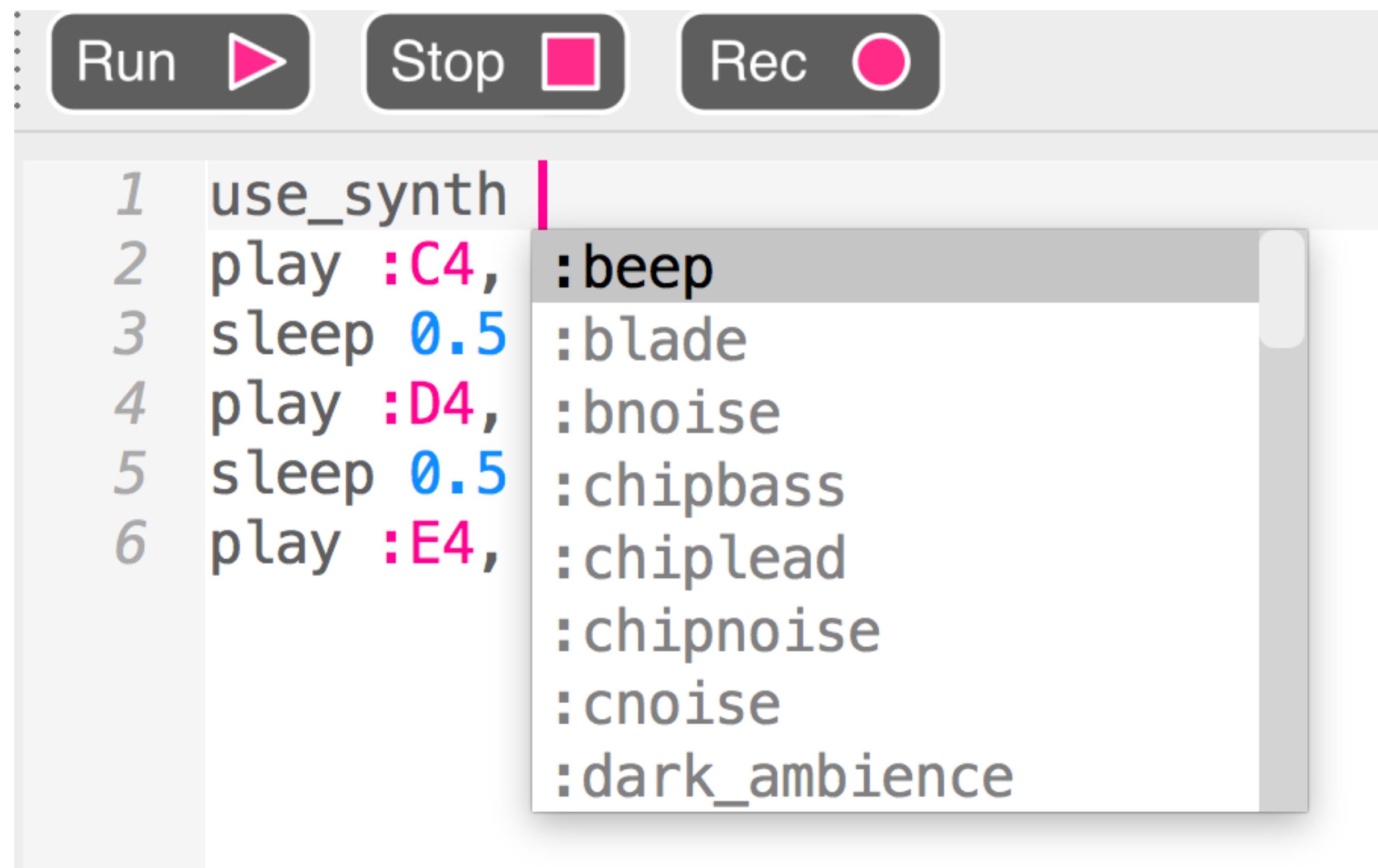
```
play :C5, amp: 1.5, pan: 0
sleep 0.5
play :Fs5, amp: 0.5, pan: -0.8
sleep 0.5
play :G5, amp: 1.0, pan: 0.8
```

Sonic Pi、プログラミング入門

- ▶ 音色を変える - シンセの指定
- ▶ これまでの音はビープ音のみ → もっと多様な音色(楽器)が用意されている!

Sonic Pi、プログラミング入門

- ▶ プログラムの始めに、「use_synth」と入力してスペースキーを入力
- ▶ :beep などいろいろなシンセ(楽器)名が表示される
- ▶ 好きなシンセを選んでみる!



Sonic Pi、プログラミング入門

- ▶ シンセの指定 - この例ではprophetを指定

```
use_synth :prophet
play :C5, amp: 1.5, pan: 0
sleep 0.5
play :Fs5, amp: 0.5, pan: -0.8
sleep 0.5
play :G5, amp: 1.0, pan: 0.8
```

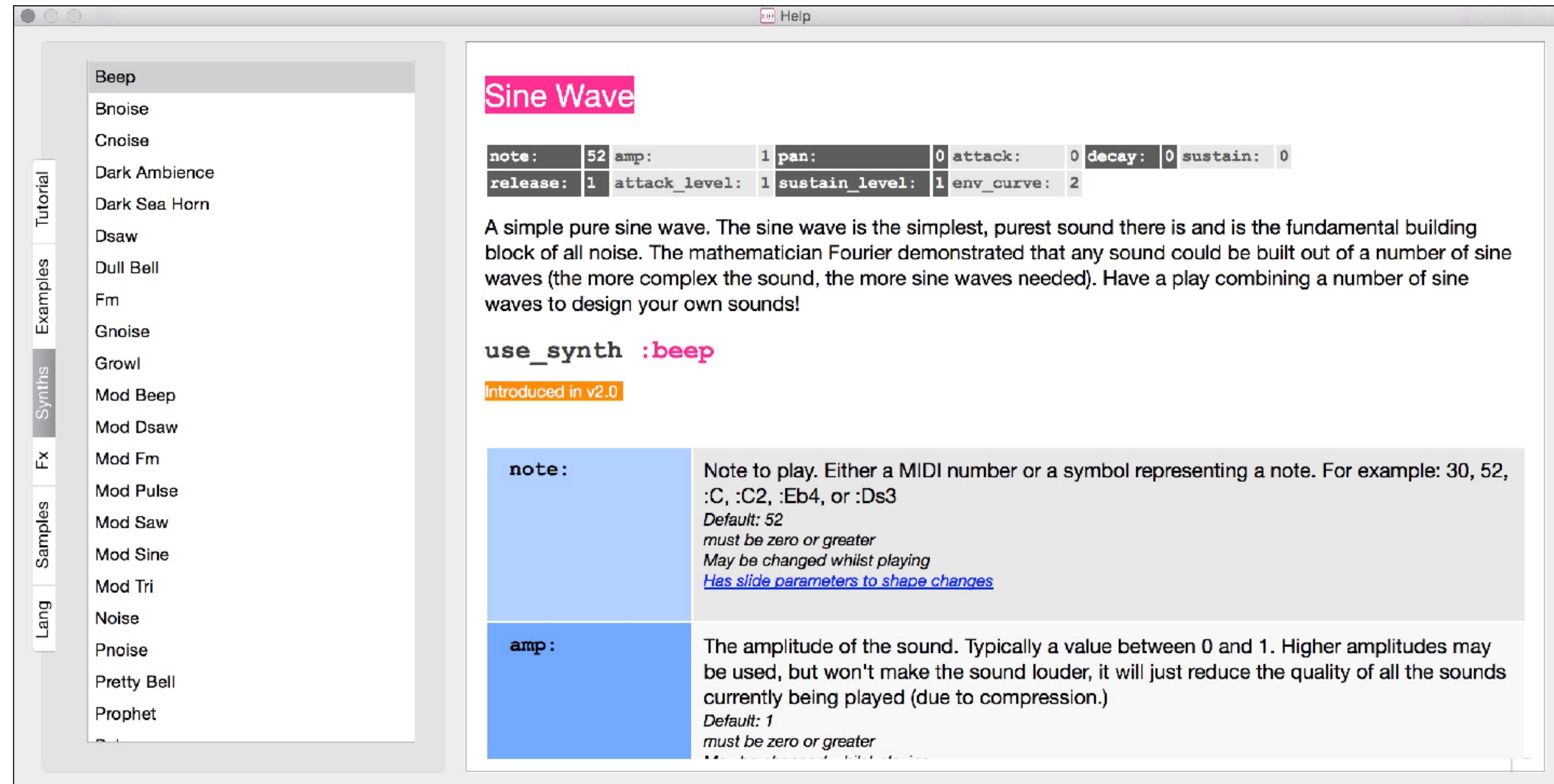
Sonic Pi、プログラミング入門

- ▶ シンセは1音ごとに切り替えることもできる

```
use_synth :prophet
play :C5, amp: 1.5, pan: 0
sleep 0.5
use_synth :zawa
play :Fs5, amp: 0.5, pan: -0.8
use_synth :saw
sleep 0.5
play :G5, amp: 1.0, pan: 0.8
```

Sonic Pi、プログラミング入門

- ▶ シンセの詳細は、Helpメニューの「シンセ」タブで確認!



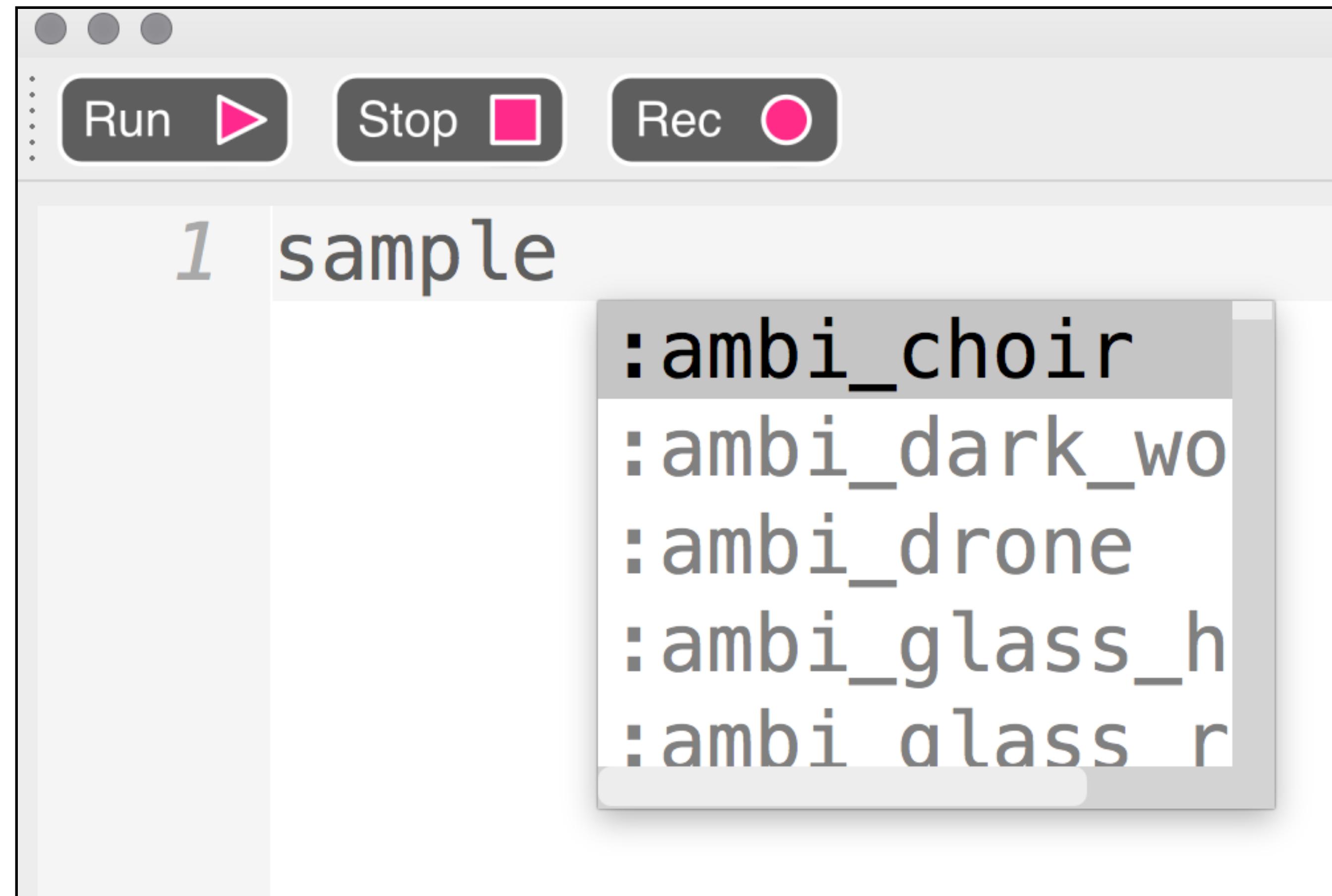
+サンプルを使う

サンプルを使う

- ▶ これまで使ってきたシンセ (:saw、:fm、:prophet) など
 - ▶ 音響合成した音 (シンセサイザーのようなもの)
-
- ▶ もう1つ別の音を出す方法
 - ▶ すでに録音された音 (サンプル) を使う
 - ▶ Sonic Piでは、多くのサンプルが用意されている

サンプルを使う

- ▶ sample と入力してスペースキーを入力 → サンプルのリストが出てくる



サンプルを使う

- ▶ 1つ選択して、Run してみる

sample : ambi_lunar_land

サンプルを使う

- ▶ シンセや複数のサンプルを同時に演奏することも可能

```
play 36
play 48
sample :ambi_lunar_land
sample :ambi_dark_woosh
```

サンプルを使う

- もし間を空けたいなら、 sleep を使う

```
sample :ambi_lunar_land
sleep 1
play 48
sleep 0.5
play 36
sample :ambi_dark_woosh
```

サンプルを使う

- ▶ シンセと同様に、ampとpanの設定が可能

```
sample :ambi_lunar_land, amp: 0.75, pan: -0.5
sleep 1
play 48
sleep 0.5
play 36
sample :ambi_dark_woosh, amp: 1.5, pan: 0.5
sleep 1
```

サンプルを使う

- ▶ rate - サンプルの再生スピードを変化させるオプション
- ▶ 例:
- ▶ rate: 1.0 - 何も指定していない時と同じ (デフォルト)
- ▶ rate: 0.5 - 再生時間が2倍に 音程は1オクターブ低く
- ▶ rate: 2.0 - 再生時間が半分に 音程は1オクターブ高く
- ▶ rate: -1.0 - 逆再生

`new_sample_duration = (1 / rate) * sample_duration`

サンプルを使う

- ▶ いろいろなサンプルで rate を試してみる

```
sample :ambi_choir, rate: 0.5
sleep 1
sample :ambi_choir, rate: 2.0
sleep 1
sample :ambi_choir, rate: -1.0
sleep 1
sample :ambi_choir, rate: 1.0
sample :ambi_choir, rate: 0.25
```

ループ (くりかえし)

ループ (くりかえし)

- ▶ くりかえし (ループ)
- ▶ loop do … end というブロックの中に入る

ループ(くりかえし)

- ▶ まずは基本パターン作成

```
use_synth :prophet
play :C4, amp: 1.5, pan: 0
sleep 0.25
play :F4, amp: 0.7, pan: -1
sleep 0.25
play :G4, amp: 1.0, pan: 1
sleep 0.5
```

ループ(くりかえし)

- ▶ プログラムの最初に「loop do」を追加、最後に「end」を追加

```
loop do
    use_synth :prophet
    play :C4, amp: 1.5, pan: 0
    sleep 0.25
    play :F4, amp: 0.7, pan: -1
    sleep 0.25
    play :G4, amp: 1.0, pan: 1
    sleep 0.5
end
```

ライブコーディング!!

ライブコーディング!!

- ▶ いよいよライブコーディングに突入
- ▶ loop do … endを変更
- ▶ live_loop :hoge do … end に

ライブコーディング!!

- ▶ 「loop do」を、「live_loop :live do」と書き換えてみる

```
live_loop :live do
  use_synth :prophet
  play :C4, amp: 1.5, pan: 0
  sleep 0.25
  play :F4, amp: 0.7, pan: -1
  sleep 0.25
  play :G4, amp: 1.0, pan: 1
  sleep 0.5
end
```

ライブコーディング!!

- ▶ プログラムを1箇所書き換えてみる
- ▶ ストップボタンを押さずに、もう一度 RUN ボタンを押す
- ▶ リズムは維持されたまま、次のサイクルで変更されるはず!!
- ▶ → ライブコーディング!!

ライブコーディング!!

- ▶ 音程、音色、リズム、何でも変更可能!!

```
live_loop :live do
  use_synth :mod_dsaw
  play :C4, amp: 1.5, pan: 0
  sleep 0.25
  play :F4, amp: 0.7, pan: -1
  sleep 0.25
  play :G3, amp: 1.0, pan: 1
  sleep 0.25
  play :G5, amp: 1.0, pan: 1
  sleep 0.25
end
```

ライブコーディングセッション!! 「In C」

ライブコーディングセッション!! 「In C」

- ▶ 全員で、ライブコーディングセッションを体験してみましょう!
- ▶ 題材は、テリー・ライリー (Terry Riley) の、「In C」 (…のようなもの)

ライブコーディングセッション!! 「In C」

- ▶ In C - 53の短かいフレーズから1つを選択、「C音」のパルスにあわせて一斉に演奏
- ▶ https://open.spotify.com/track/2q5NfHsScylyJxLAAMA_Pbm

in C.

The musical score consists of five staves of music, each containing ten numbered measures. The measures are numbered sequentially from 1 to 53. The music is written in common time with a treble clef. The notes are primarily eighth and sixteenth notes, with some quarter notes and rests. Measure 1 starts with a single eighth note followed by a sixteenth note. Measure 2 has a sixteenth note followed by an eighth note. Measures 3 through 6 show various patterns of eighth and sixteenth notes. Measures 7 through 10 continue the rhythmic patterns. Measures 11 through 15 show more complex patterns, including measure 13 which features a sixteenth-note run. Measures 16 through 20 show further variations. Measures 21 through 25 show a mix of eighth and sixteenth-note patterns. Measures 26 through 30 show a continuation of the patterns. Measures 31 through 35 show a mix of eighth and sixteenth-note patterns. Measures 36 through 40 show a continuation of the patterns. Measures 41 through 45 show a mix of eighth and sixteenth-note patterns. Measures 46 through 50 show a continuation of the patterns. Measures 51 through 53 show a final set of patterns.

1. 2. 3. 4. 5. 6.
7. 8. 9. 10.
11. 12. 13. 14. 15.
16. 17. 18. 19. 20. 21.
22. 23. 24.
25. 26. 27. 28.
29. 30. 31. 32. 33. 34.
35.
36. 37. 38. 39. 40. 41. 42.
43. 44. 45. 46. 47.
48. 49. 50. 51. 52. 53.

© 1964
Terry Riley
© 1989
Celestial Harmonies

ライブコーディングセッション!! 「In C」

- ▶ 参考: テリー・ライリー《In C》のざっくり演奏方法
- ▶ <https://www.youtube.com/watch?v=Xiy2Zby5kyQ>



ライブコーディングセッション!! 「In C」

- ▶ 今回は、もっとざっくりとしたシステムで
- ▶ Cの音に調和するようなフレーズをライブコーディングで作成し演奏していく
- ▶ フレーズ、音色、リズムは基本自由、ただしテンポだけは合わせて!

- ▶ やってみましょう!!

ライブコーディングセッション!! 「In C」

- ▶ まずは、このパターンから → 周囲の音を聞きながら変化させていきましょう!

```
live_loop :live do
  use_synth :piano
  play :C4, amp: 1.0, pan: 0
  sleep 0.25
end
```

ワークショップの今後の展開

ワークショップの今後の展開

- ▶ 今回は「イントロダクション」
- ▶ 2017年から、より本格的なワークショップシリーズを開始!!
- ▶ 本格的なライブコーディングをマスター
- ▶ ワークショップ参加者によるイベントを開催 (願望) !!

ありがとうございました。