

自分だけの世界を創る クリエイティブコーディング

10/5 YAPC::Hakodate 2024

@chobishiba



こしば 🌱 chobishiba

SmartBank, Inc.

サーバーサイドエンジニア

 @chobishiba

 @ksbmyk



YAPC 函館

2024 OPEN THE FUTURE

本日登壇する
スマートバンク
メンバー

キーノートセッション

Track A 17:20～

ライトニングトーク

Track A 16:40～



@moznion
Taiki Kawakami

プロファイラ開発者とする
「推測するな、計測せよ」

Track C 11:05～

@osyoyu
Daisuke Aritomo



クレジットカードを
製造する技術

Track C 15:50～

@yutadayo
Yuta Horii



ライトニングトーク

Track A 16:40～

@chobishiba
Miyuki Koshiba



お時間が合えば ぜひご覧ください！

自分だけの世界を創る
クリエイティブコーディング

突然ですがここに1行のPerlのコードが



```
print map { rand() < 0.5 ? "\" : "/" } (1..1000); print "\n";
```

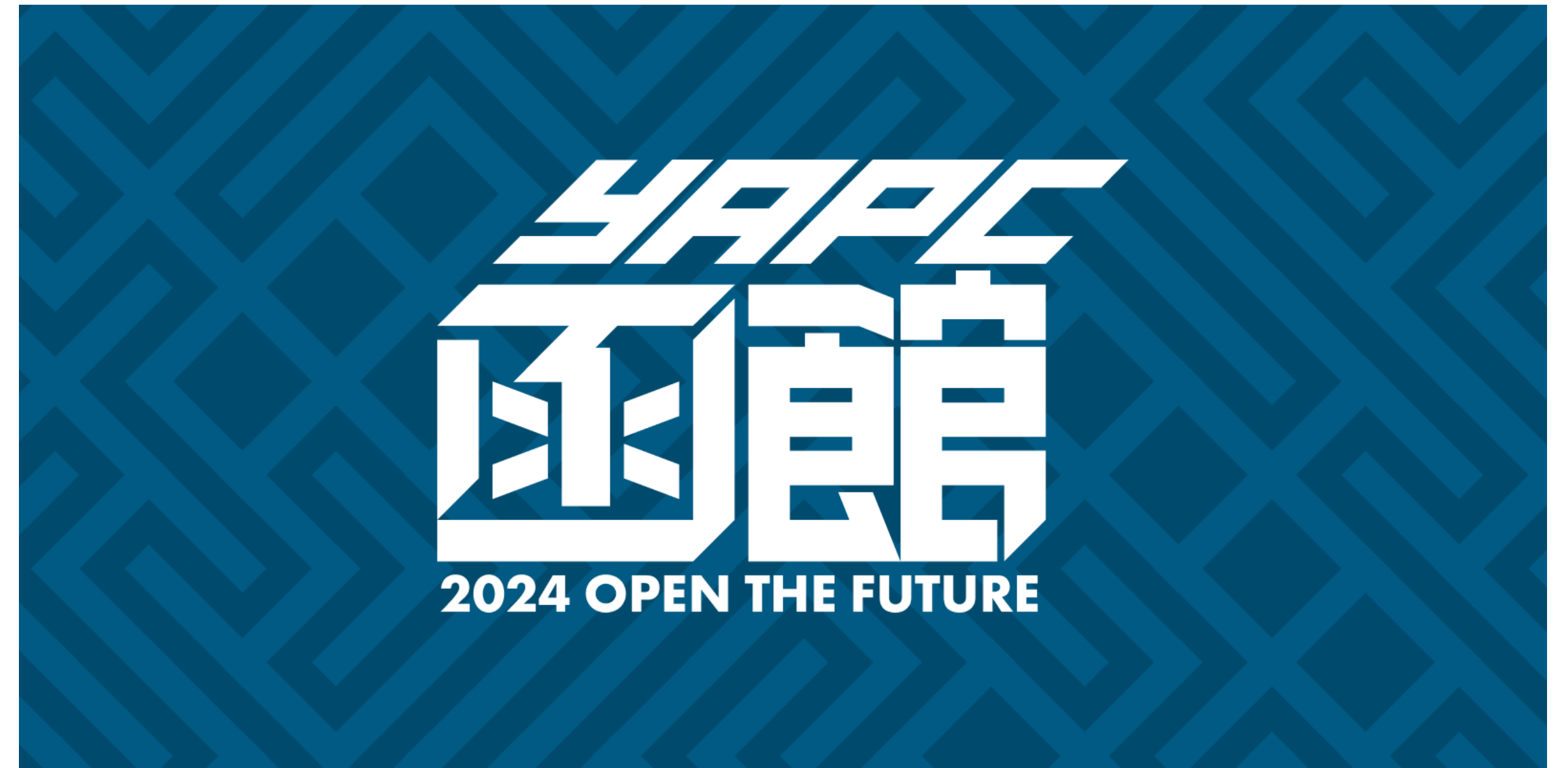
```
→ ~ perl yapc.pl
```



```
➔ ~ perl yapc.pl
```

どこかで見たことはありませんか？





▲背景の柄



```
10 PRINT CHR$(205.5+RND(1)); : GOTO 10
```

- 1980年代に発表されたワンライナー BASICコード
- ランダムに「／」と「＼」の文字を画面に出力すると迷路のようなパターンができる



10print <https://10print.org/>

画像として書き出すこともできます

```
use strict;
use warnings;

use Imager;

my $img_width = 400;
my $img_height = 400;
my $cell_size = 40;
my $line_width = 8;

my $img = Imager->new(
    xsize => $img_width,
    ysize => $img_height,
    channels => 3
);

my $background_color = Imager::Color->new('#004b6d');
$img->box( color => $background_color, filled => 1 );
my $line_color = Imager::Color->new('#0174a9');

for (my $x = 0; $x < $img_width; $x += $cell_size) {
    for (my $y = 0; $y < $img_height; $y += $cell_size) {
        my $draw_slash = int(rand(2));

        if ($draw_slash == 0) {
            # "\"
            my @points = (
                [$x - ($line_width / 2), $y + ($line_width / 2)], # 左上
                [$x + ($line_width / 2), $y - ($line_width / 2)], # 右上
                [$x + $cell_size + ($line_width / 2), $y + $cell_size - ($line_width / 2)], # 右下
                [$x + $cell_size - ($line_width / 2), $y + $cell_size + ($line_width / 2)] # 左下
            );
            $img->polygon(points => \@points, color => $line_color, filled => 1);
        } else {
            # "/"
            my @points = (
                [$x + $cell_size + ($line_width / 2), $y + ($line_width / 2)], # 右上
                [$x + $cell_size - ($line_width / 2), $y - ($line_width / 2)], # 左上
                [$x - ($line_width / 2), $y + $cell_size - ($line_width / 2)], # 左下
                [$x + ($line_width / 2), $y + $cell_size + ($line_width / 2)] # 右下
            );
            $img->polygon(points => \@points, color => $line_color, filled => 1);
        }
    }
}

$img->write( file => '10print.png' ) or die $img->errstr;
```



<https://gist.github.com/ksbmyk/7b5a0397cd1f3b7db0da2aa4f33516fc>

参考 <https://www.nekonotechno.com/nekopress/?p=7166>

- アルゴリズムで作るアート作品
 - クリエイティブコーディング
 - ジェネラティブアート

- 「ジェネラティブアート（Generative Art）」は、コンピュータのアルゴリズムによって生成されるアート作品の総称です。
- 主な特徴として**計算性・自律性・ランダム性**が挙げられます。
- 作品の主題（テーマ）は、物理現象や絵画的技法のシミュレーション、計算によってはじめて可能になる表現、人工と自然の間での有機的な表現を追求するアートなど、**さまざま**です。

一般財団法人ジェネラティブアート振興財団 <https://generativeart.or.jp/about>

- 平たくいうとコードでお絵描き
- 静止画だけでなく、動画やインタラクティブなもの、実行するたびに結果が異なるような作品も作れる

- ソフトウェア開発

- 冪等性を求められることがほとんど
- 何かの問題解決のために行う

- クリエイティブコーディング

- 積極的に偶然性を取り入れる
- ただ楽しむために作る

プログラミングの1つの側面、1つの分野

- Processing(Javaベース)
- p5.js(JavaScriptのライブラリ)
- openFrameworks(C#ベース)
- Touch Designer(ノードプログラミング)



Webブラウザで動く と 共有しやすい



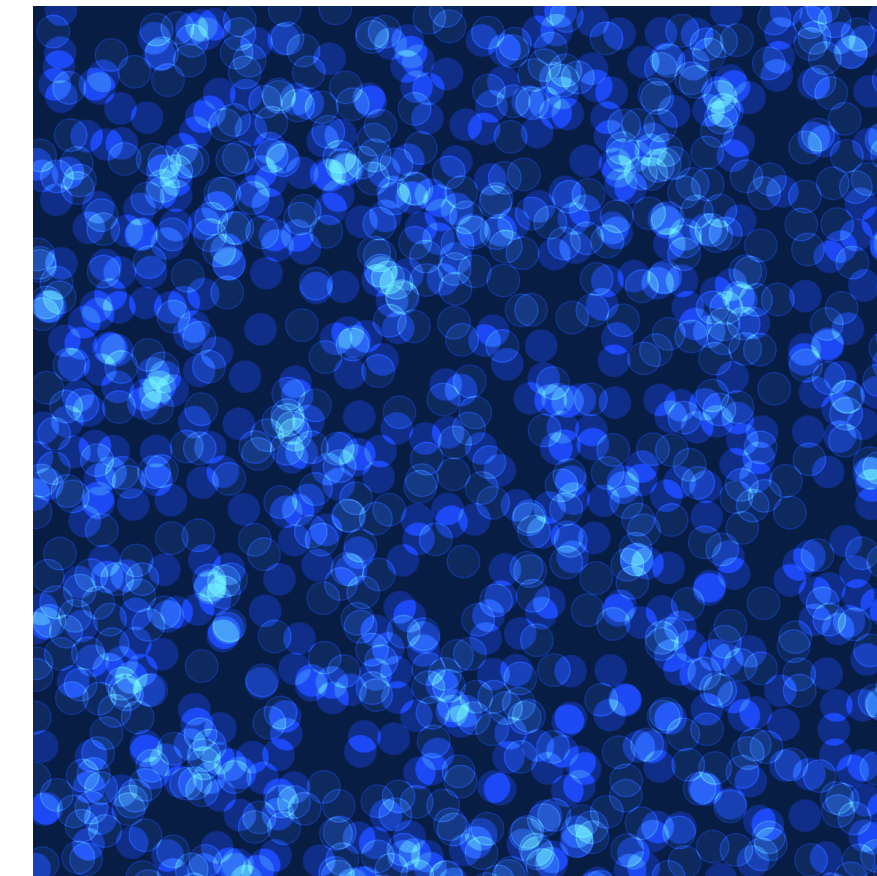
```
def setup
  createCanvas(400, 400)
  @grid_size = 20
  rectMode(CENTER)
  noLoop()
end

def draw
  background("#004b6d")
  rect_height = 7
  rect_width = @grid_size * 1.7
  fill("#0174a9")
  noStroke()

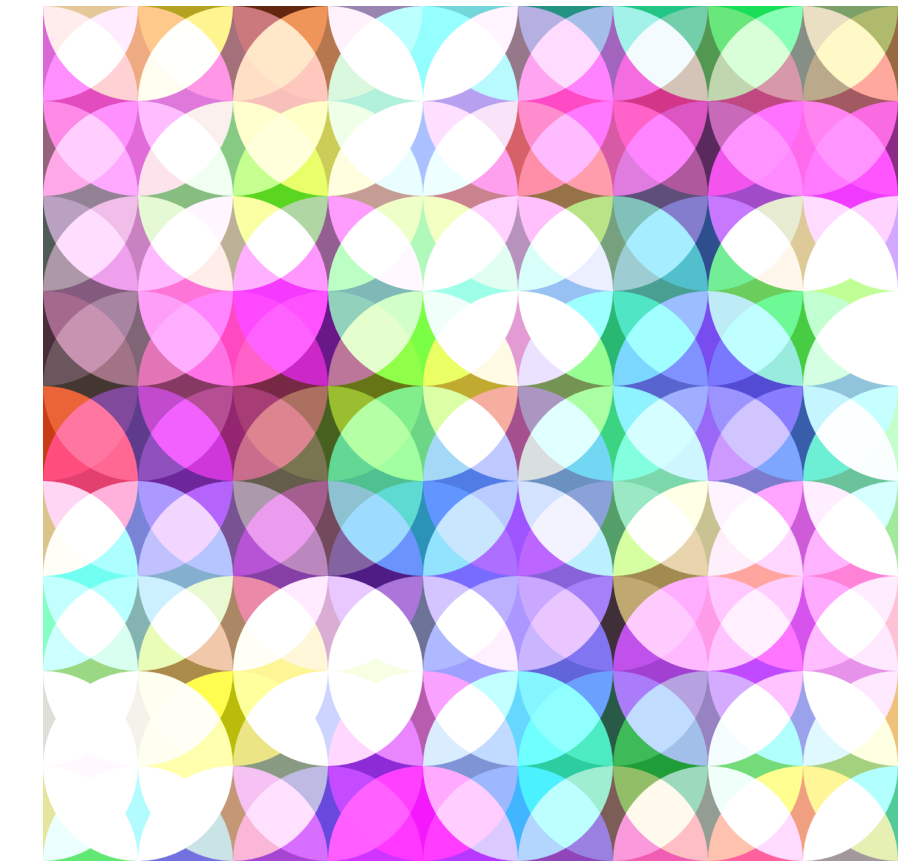
  (@grid_size / 2...height).step(@grid_size) do |y|
    (@grid_size / 2...width).step(@grid_size) do |x|
      push
      translate(x, y)
      if rand < 0.5
        rotate(PI / 4)
      else
        rotate(-PI / 4)
      end
      rect(0, 0, rect_width, rect_height)
      pop
    end
  end
end
```

p5.js+Wasm+Ruby
(めずらしいが
好きな言語を使ってやるのが楽しい)
<https://x.gd/FMIcU>

- 最初は単純なものから
- 特定の図形だけ(線だけ、円だけ)で描けるものを考えてみる
- 推しのカラーで作る！
- このアルゴリズム使いたい！



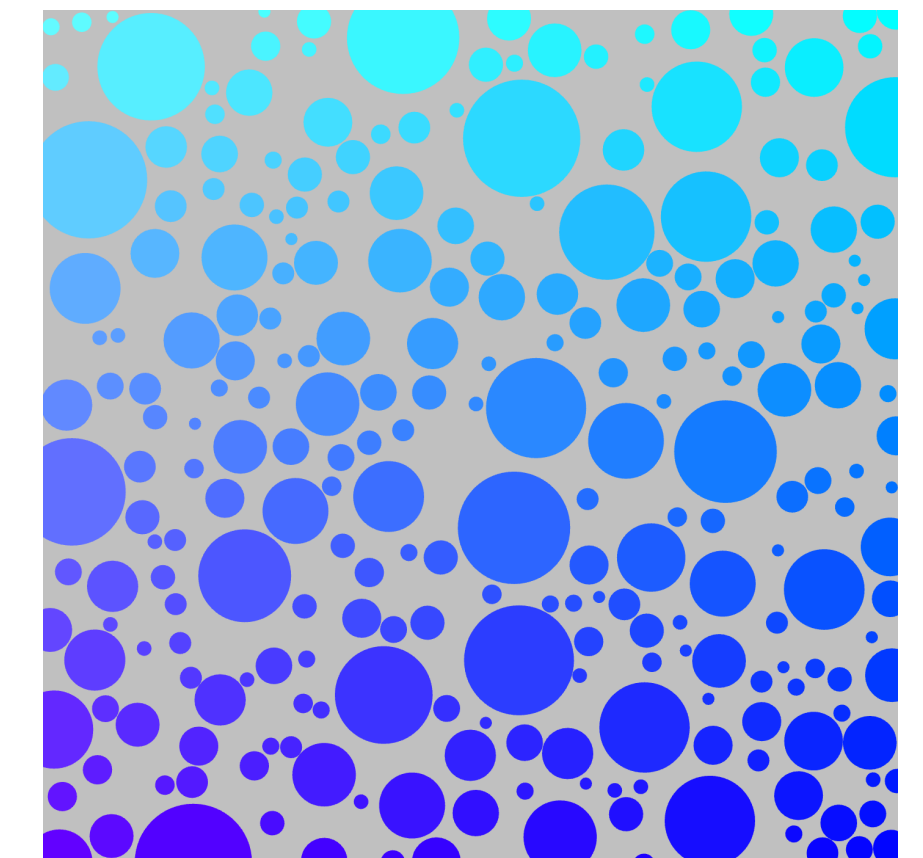
ランダム



縦横しきつめる



イラスト



サークルパッキング

円をどういう規則で並べて何色にするかの違い→

気に入ったものができたら別verを作ってみる

```
def setup
  createCanvas(600, 600)
  background("#335fa6")
  noLoop
end

def draw
  stroke(255)
  strokeWeight(10)
  noFill
  arabesque_pattern

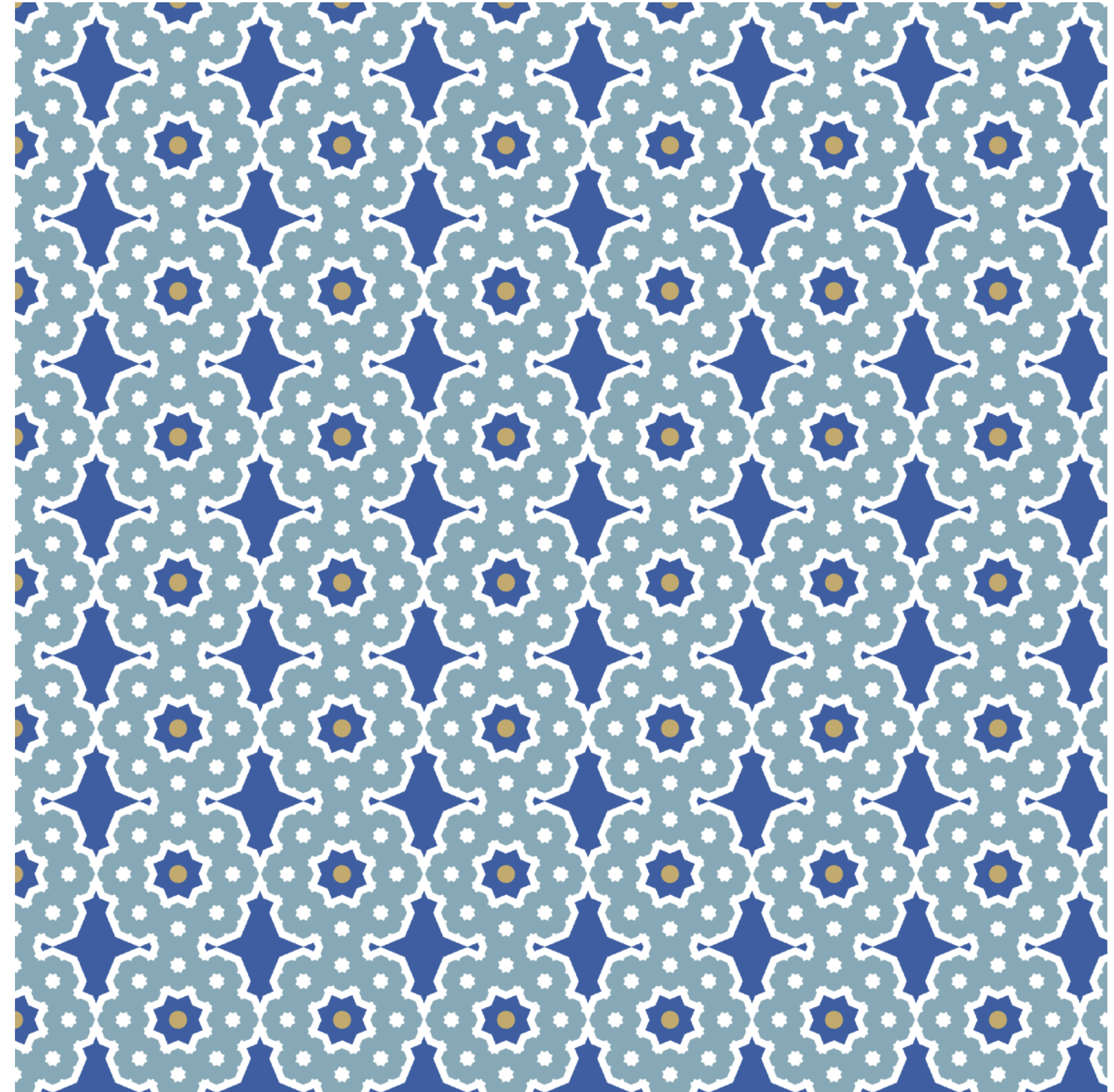
  stroke("#7eaab7")
  strokeWeight(2)
  noFill
  arabesque_pattern

  noStroke
  fill("#c7a964")
  circle_pattern
end

def arabesque_pattern
  (0..8).step do |i|
    (0..8).each { |j| arabesque(90 * i, 80 * j, 100, 8, 5) }
  end
  arabesque(90 * i, 80 * j, 100, 8, 5)
end

def circle_pattern
  (0..8).step do |i|
    (0..8).each { |j| circle(90 * i, 80 * j, 10) }
  end
  circle(90 * i, 80 * j, 10)
end

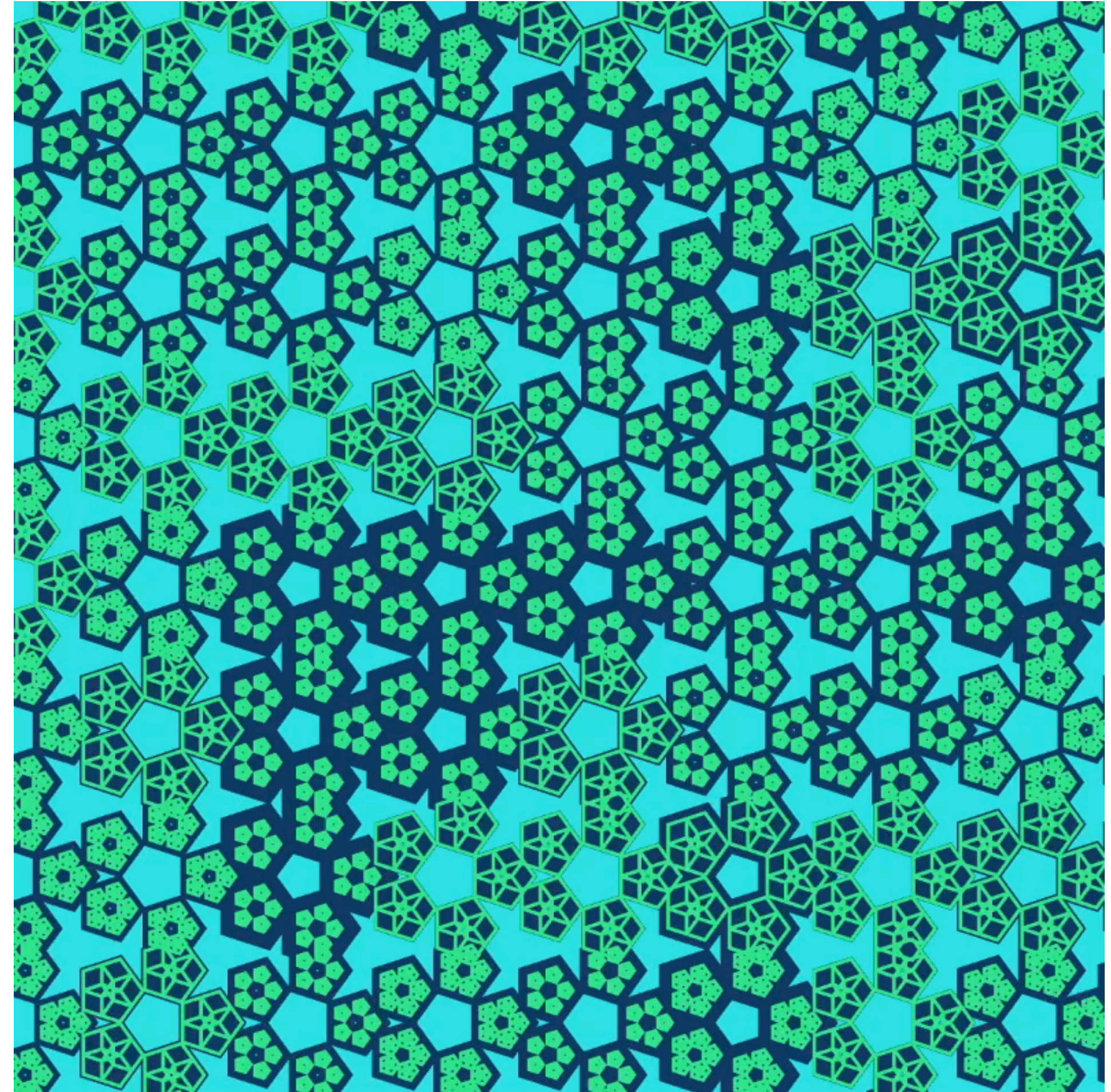
def arabesque(x, y, radius, sides, depth)
  return if depth == 0
```



再帰を使ってパターンを並べる

再帰の深さをランダムに
色もランダムにして、動くように
できたぞ！

結構ベタ書きになってしまったところ
も…よしリファクタしよう



ん ? ? ?

```
def setup
  createCanvas(600, 600)
  frameRate(0.5)
  colorMode(HSB, 360, 100, 100)
end

def draw
  colors = set_color
  background(colors[:background])
  noFill

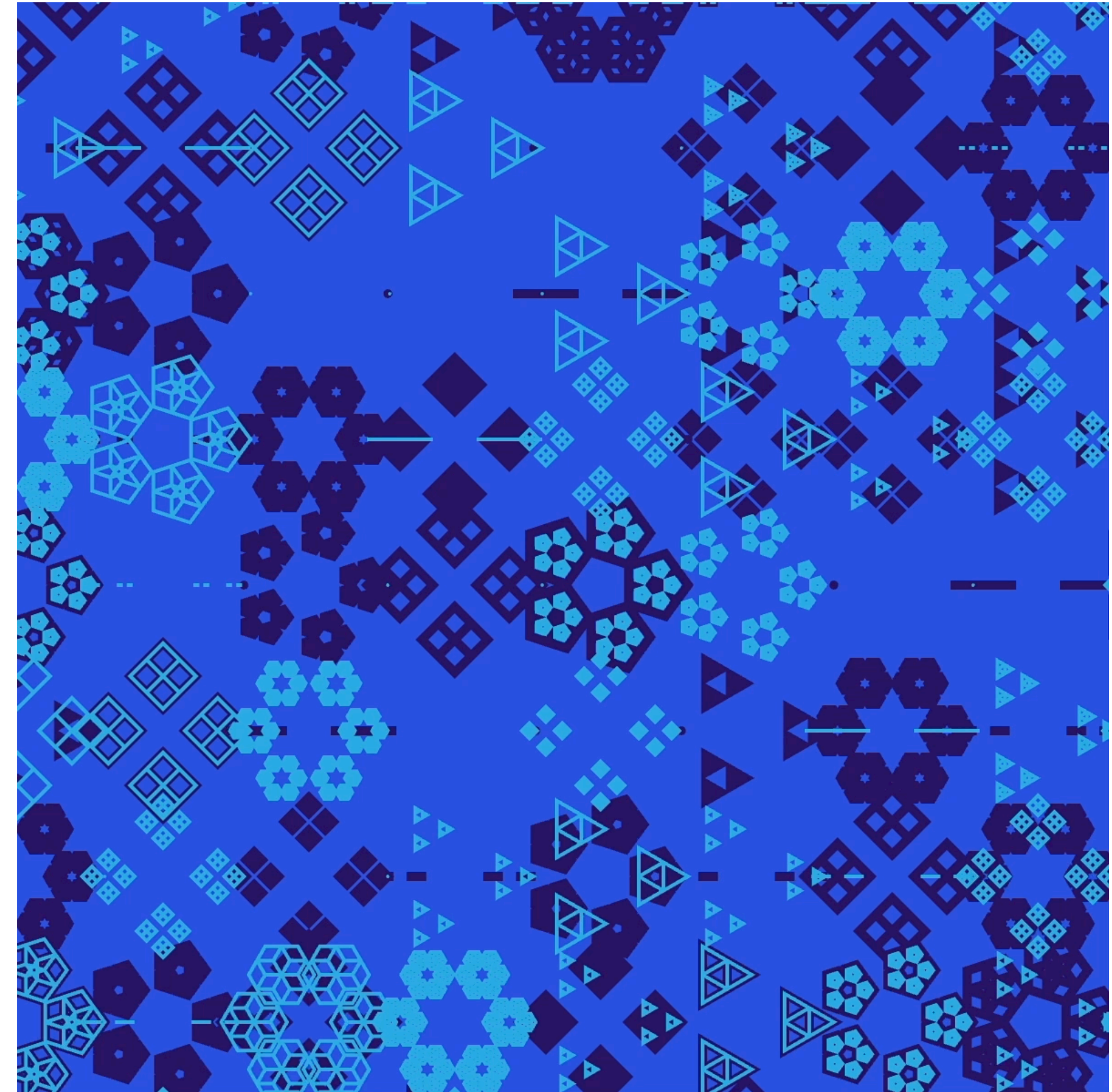
  draw_pattern(colors[:stroke1], 5)
  draw_pattern(colors[:stroke2], 2)
end

def draw_pattern(color, weight)
  stroke(color)
  strokeWeight(weight)
  (0..8).each do |i|
    (0..8).each do |j|
      geometric(80 * i, 80 * j, 100, rand(3..5), rand(1..6))
    end
  end
end

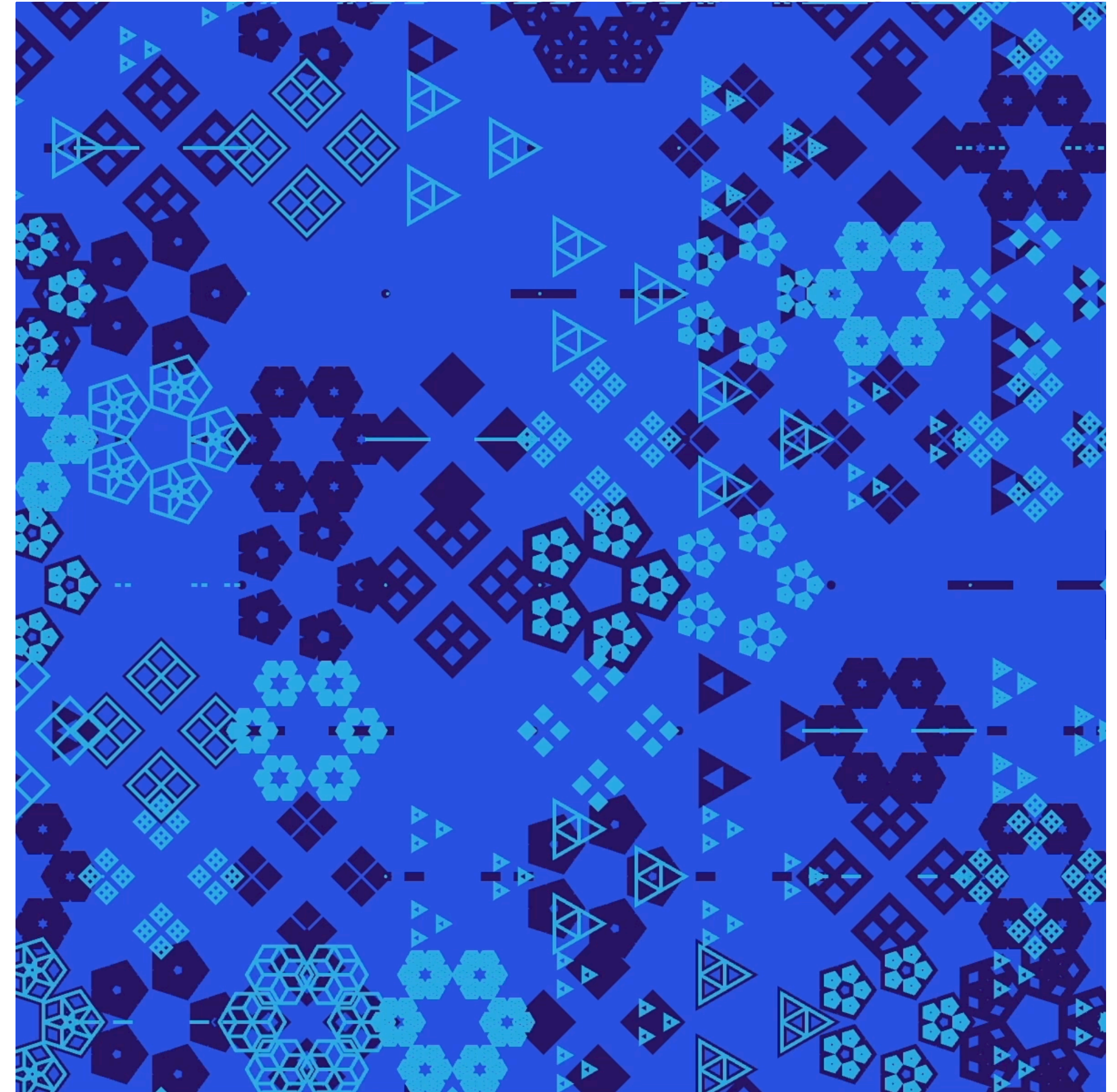
def geometric(x, y, radius, depth, num_sides)
  return if depth.zero?

  beginShape
  num_sides.times do |i|
    angle = map(i, 0, num_sides, 0, TWO_PI)
    x_offset = x + cos(angle) * radius
    y_offset = y + sin(angle) * radius
    vertex(x_offset, y_offset)

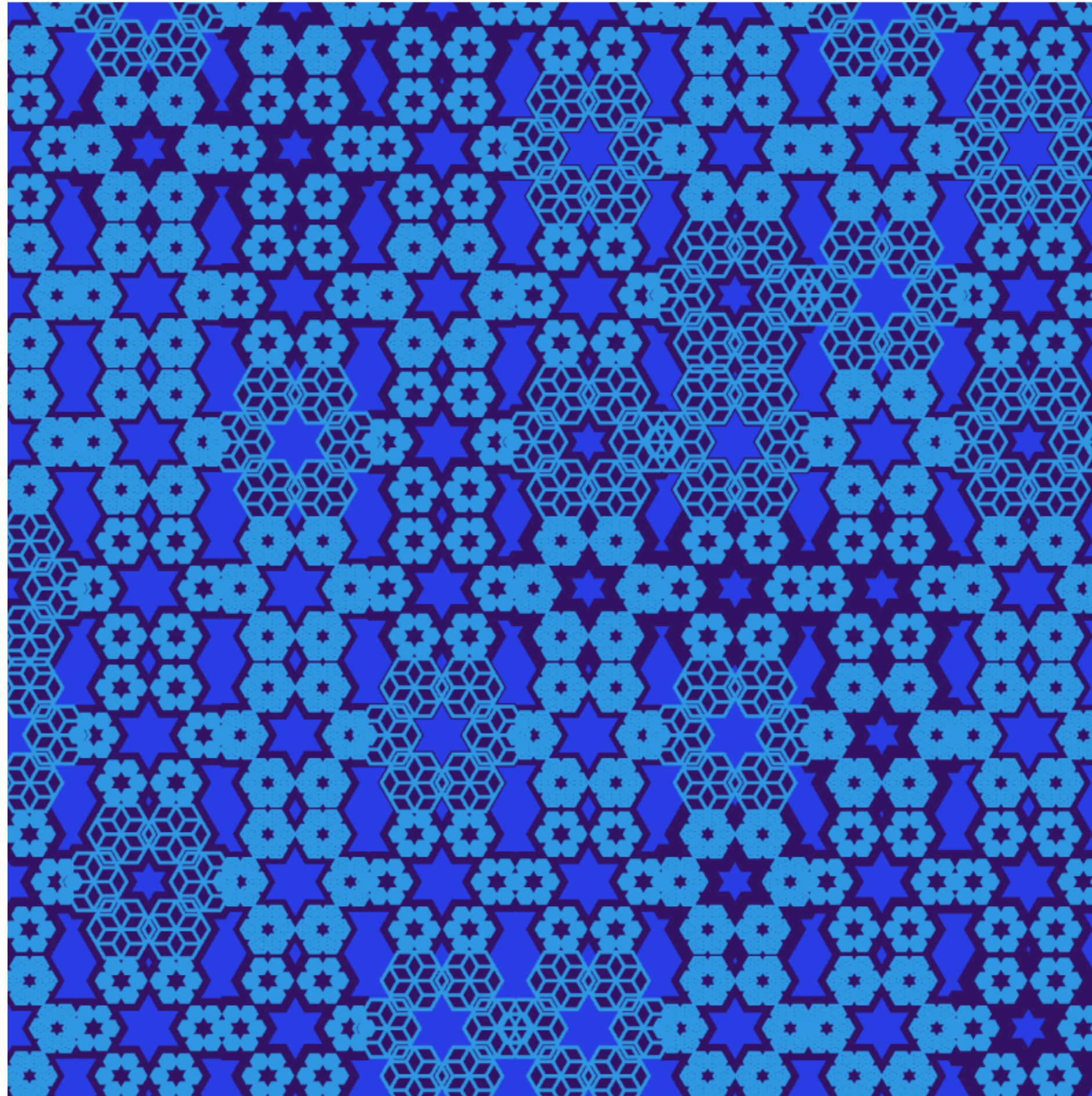
    next_radius = radius * 0.3
    next_x = x + cos(angle) * next_radius
    next_y = y + sin(angle) * next_radius
    geometric(next_x, next_y, next_radius, depth - 1, num_sides)
  end
end
```



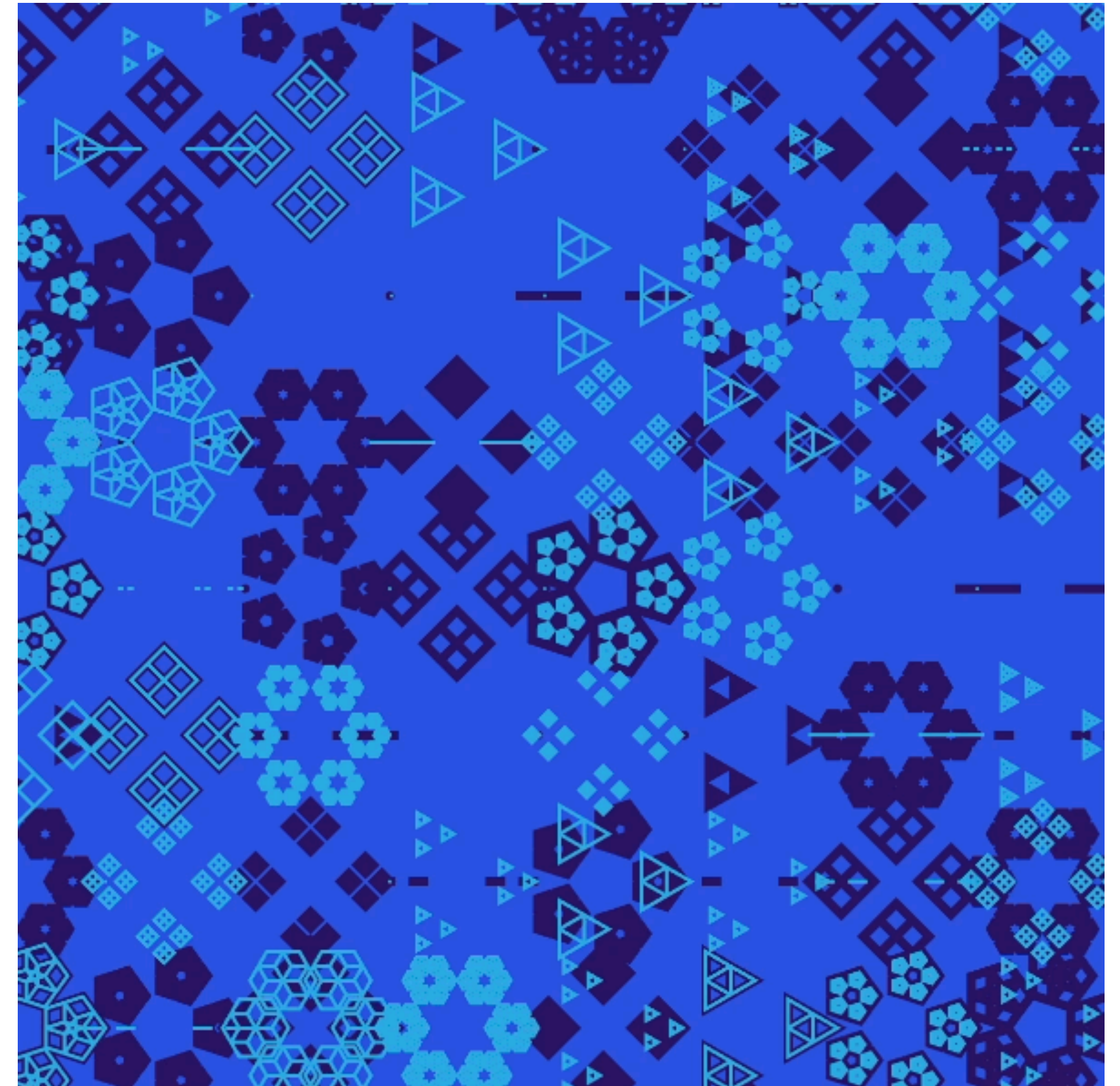
- やりたかったこと
 - 多角形の辺の数を最初に1回ランダムで決める
 - 再帰の深さは毎回ランダム
- やってしまったこと
 - 辺の数を毎回ランダムで決める
 - 再帰の深さは毎回ランダム



これはこれでいいのでは？



ちょっと窮屈・単調



ちょうどいいランダム具合

- 思ってたのと違う結果になるコードを書いてしまうことはよくある
- でも意外な発見があって自分のおっちょこちょいすらナイスとなる
- 間違いはない
- すべて「それも一つの形」
- 自分の好きにする



- 間違いがない代わりに正解もない
- どちらがよいか、何がよいか
- 全て自分で決める



- 普段仕事だと「問題」を「解決」する
 - = 目指すことは割と明確（実現は難しいことも多いけど）
- でも「好きに」ってなんだ？
 - 自分はどうしたかったっけ？
 - 途方に暮れることも



- それでもちょっとでも自分好きな方を選ぶ
- 選びたいものを選ぶ
- 自分の好き勝手に、思い通りに
- やっていると何かが回復してくる感覚



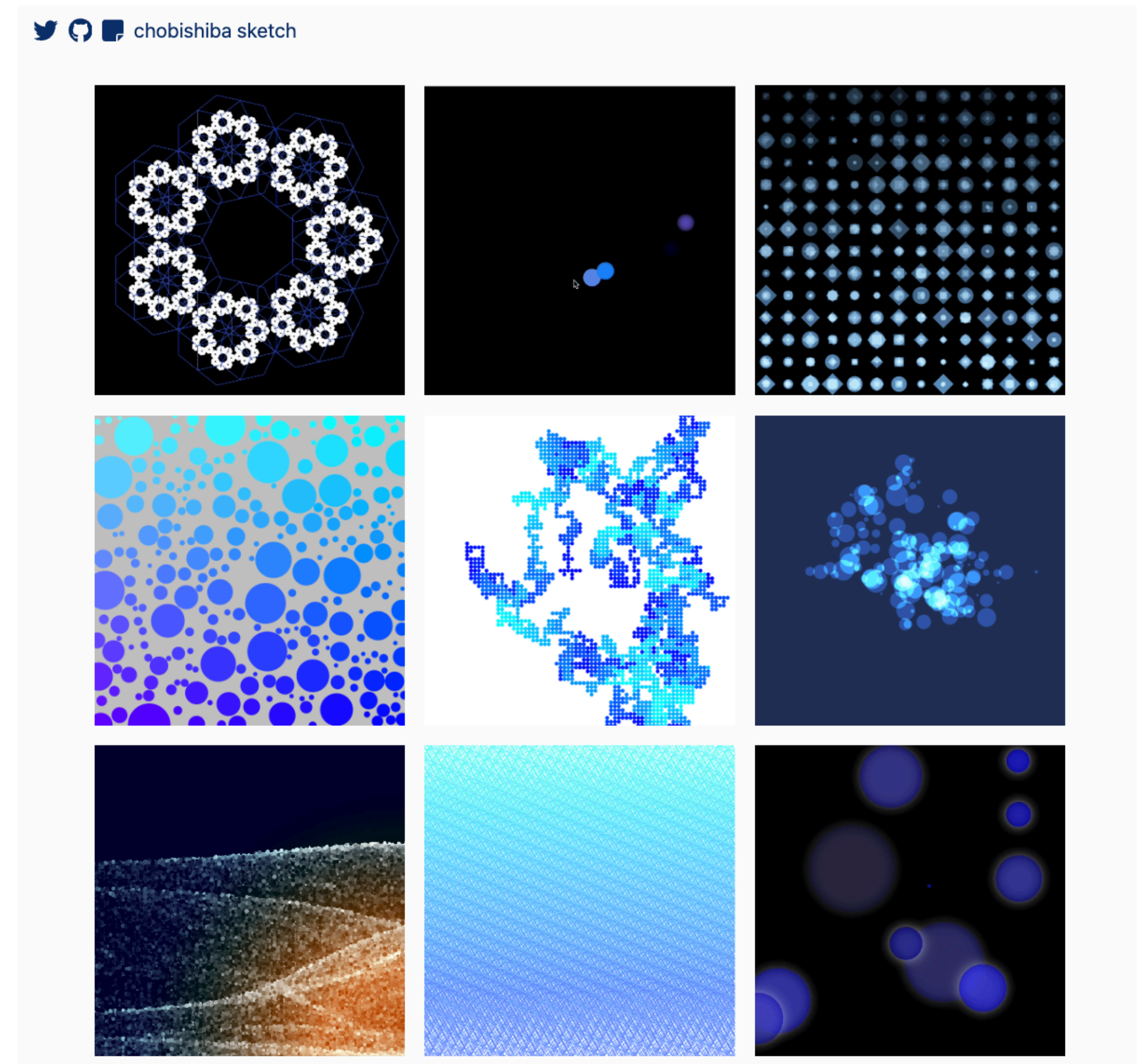
- 自分のための世界を自分で創るという行為
- 好きにする＝自分を優先する＝自分を大事にする



- コードを書く→動く→楽しい！のループを短く回せる
- 描画される＝常に何らかのアウトプットがある
 - 達成感がある
- 前に作ったものを改変して別なものにもできる
 - 盆栽のようにちまちま育てていくこともできる



- プラットホームで公開
 - OpenProcessing, NEORT...
- SNSで公開
 - X, Instagram...
- 個人サイトを作る



今日紹介した作品のコードもあります→ <https://ksbmyk.github.io/sketch/>

- グッズにしても楽しい
- 技術同人誌の表紙にもいい



pixivFactory <https://factory.pixiv.net>

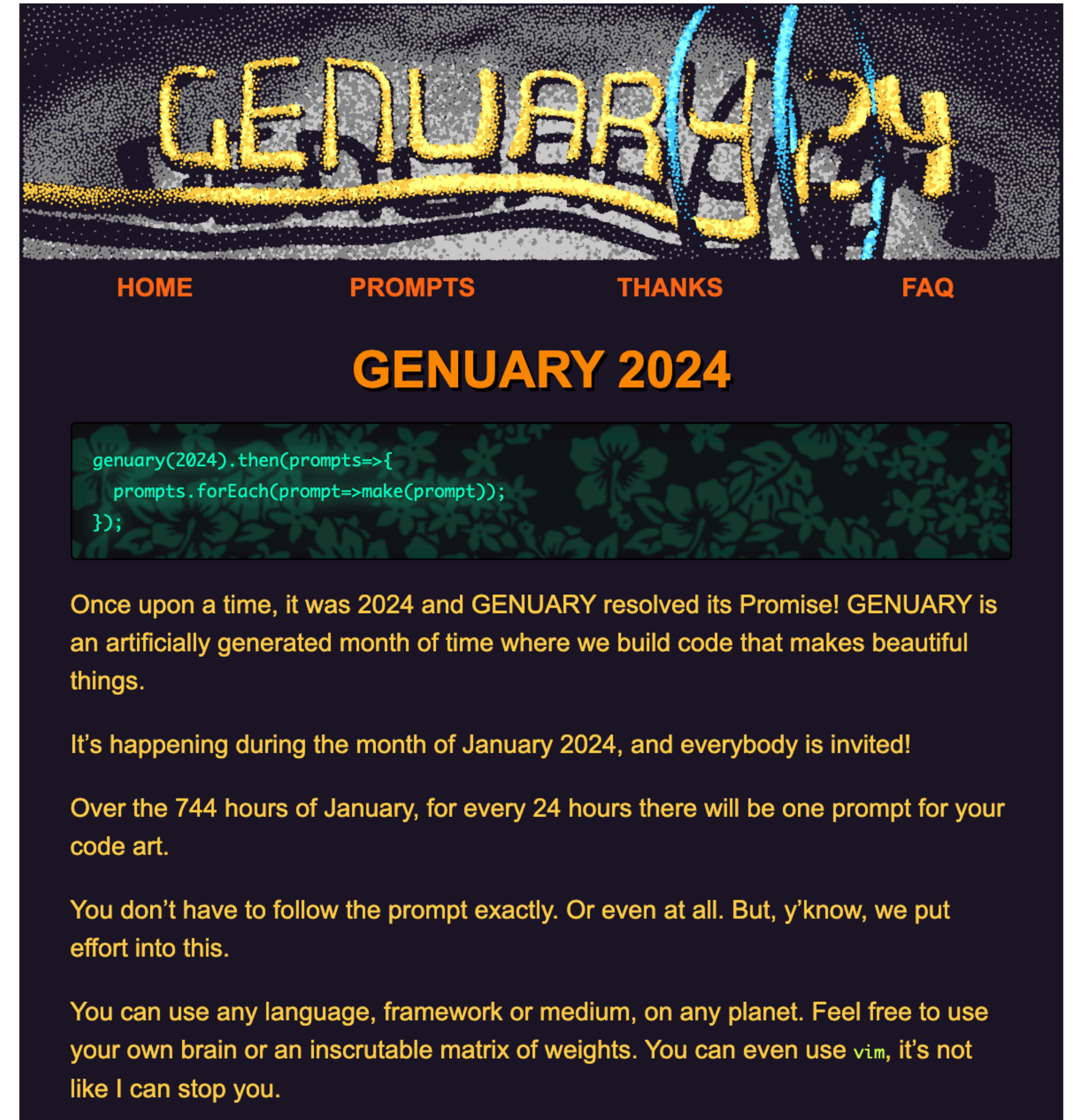
- クリエイティブコーディングステッカー配ってます！
- もしよかったらこのあと声かけてください



- ひとり息抜きにちまちまやってた
- ちょっと違うことがしてみたくなった

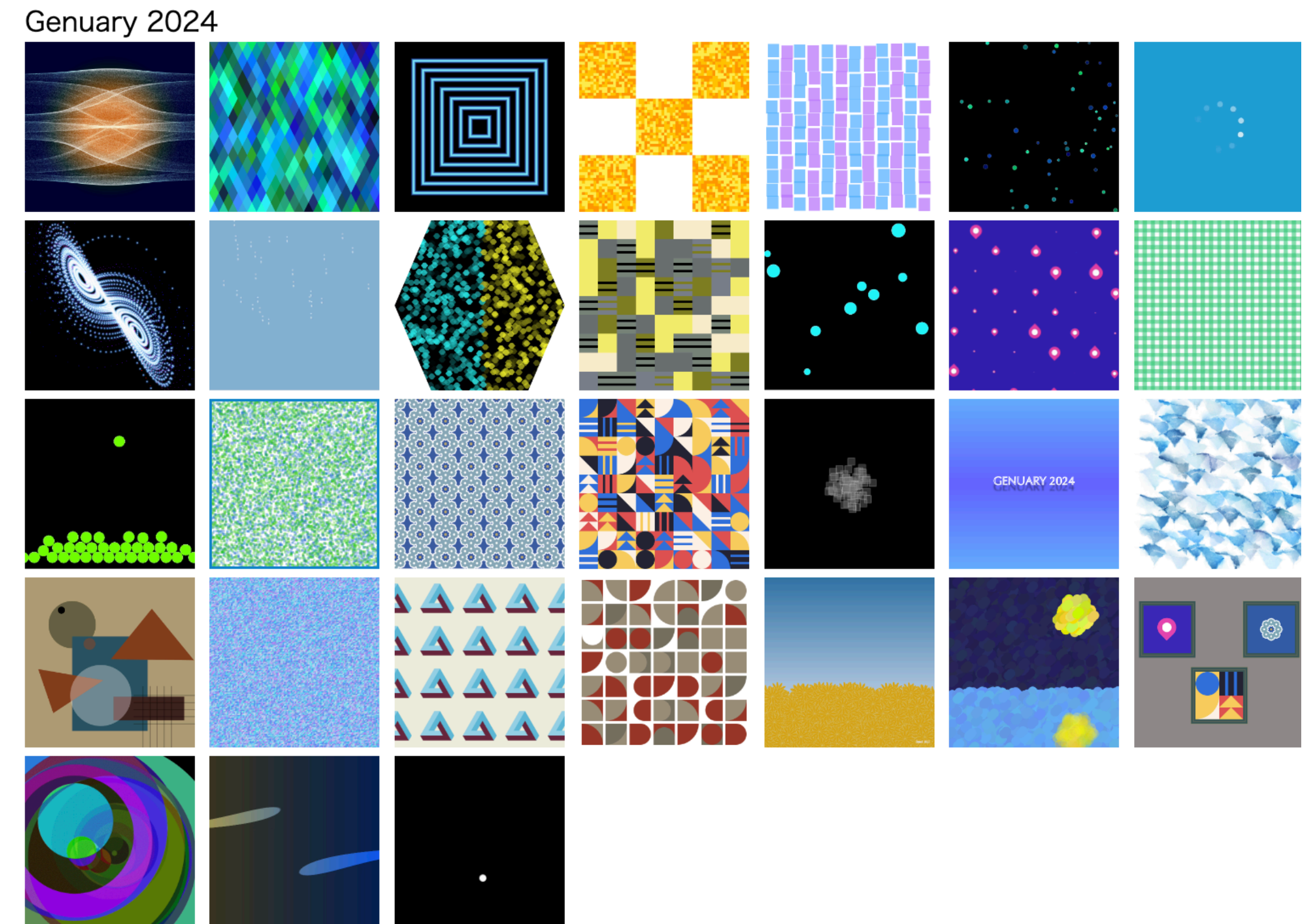


- Genuary (= January + Gen Art)
- お題に沿ってコードで作品作る
 - 1ヶ月間毎日お題が出る
 - 作って公開する（任意）
- 世界中のクリエイターが参加



<https://genuary.art/>

- 毎日作って投稿続けて31日間完走
- 仕事の日も、飲み会の日も
- お題のおかげで手癖じゃ作らないものも作れた
- 作れるものの幅が広がった



- 世界中の参加者とSNS上でゆるく交流ができた
- 話す言語も、作品作りに使う言語も違う人たち
- でも同じことにチャレンジした仲間
- 世界が広がった



- この体験をLTネタにした
- 「いつかどこかでワークショップしたい」と書いた

To be continue...

どうやってやるかを話出すところからさらに長くなるので詳しくは以前の発表資料とか、このあと直接聞いてもらえれば

(いつかどこかでワークショップもやりたい)

クリエイティブコーディングで作ったステッカーも配ってるので気軽に声かけて下さい！



<https://speakerdeck.com/chobishiba/creative-coding-starting-with-ruby>



SMART BANK

41

<https://speakerdeck.com/chobishiba/enjoy-programming>

- Waffle Festival 2024
 - 学生向けカンファレンス
- プログラミング入門WSとしてクリエイティブコーディングWSの依頼
- アウトプットとして絵が出るのは興味を持ってもらやすい
- その後中学校でも行う機会が

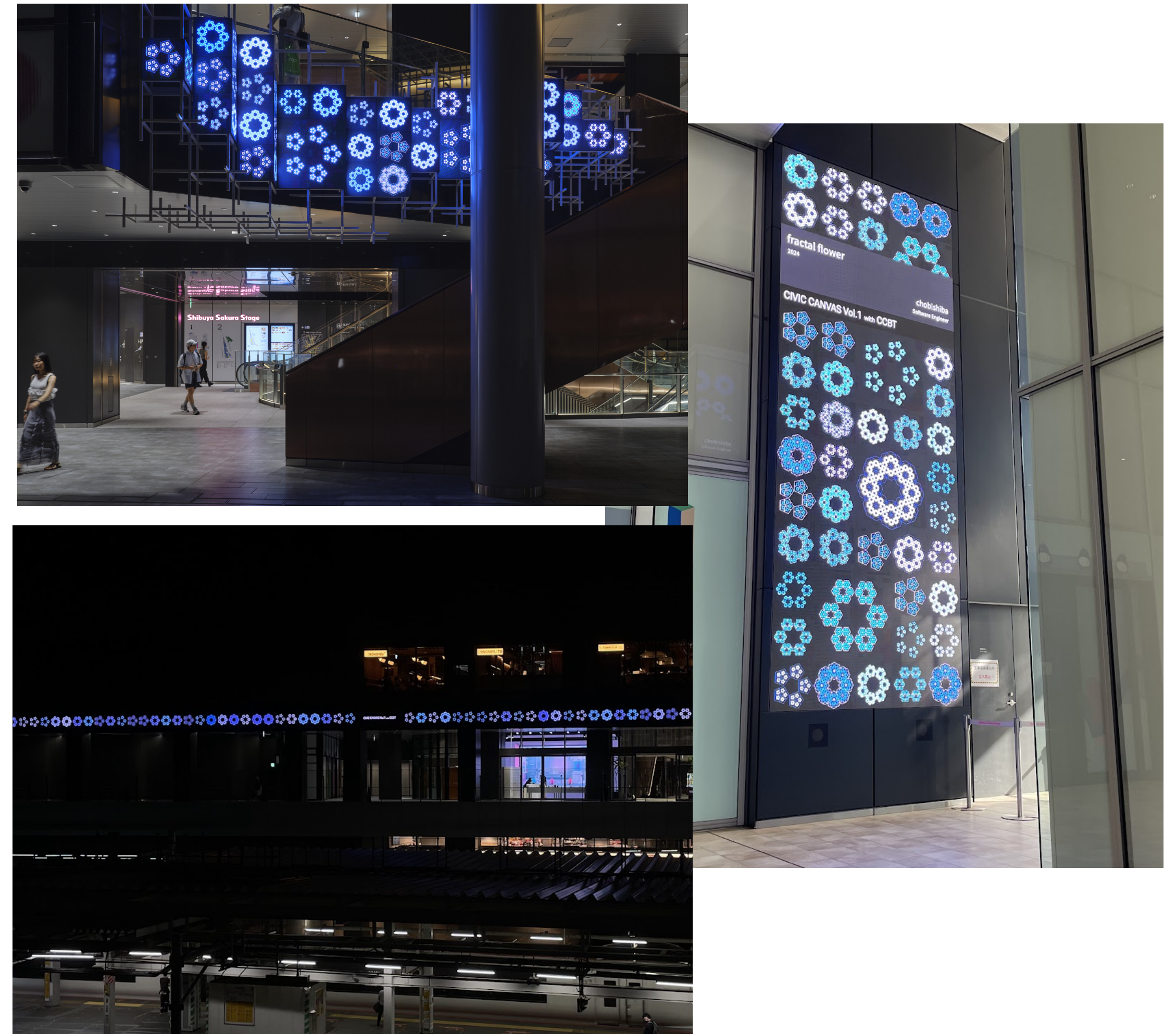


<https://waffle-waffle.org/2024/07/11/wafflefestival2024/>

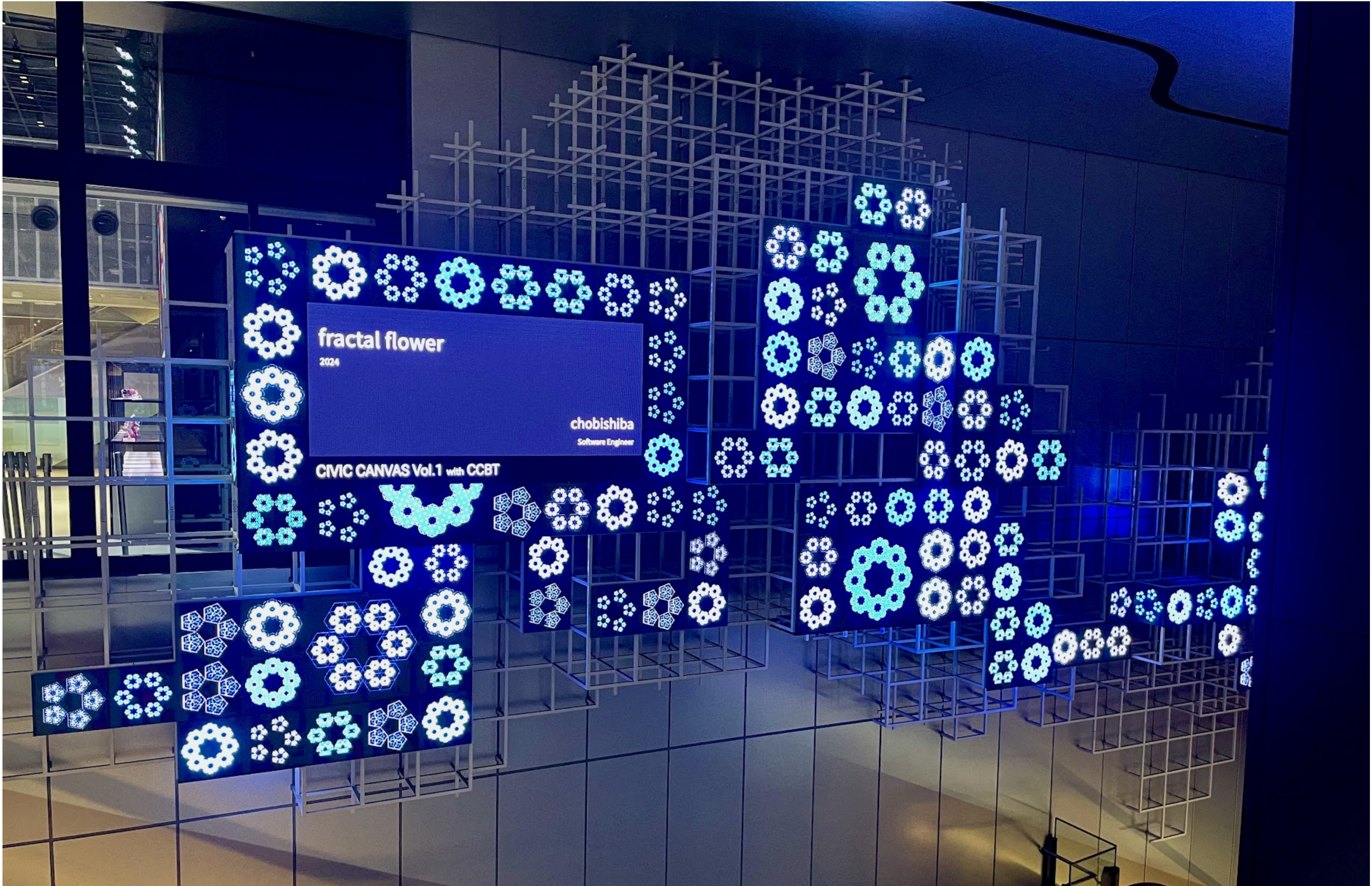
- 好きだといっていると情報が入ってくる
- 目につきやすくなるし、教えてももらえる
- アウトプットしてるとどのくらい作れるか知ってもらえる



- 東京の渋谷に新しくできた商業施設で展示できる機会
- 全力で応募、選考通過
- 2週間施設内の4箇所のメディアファサードで上映してもらった



<https://ccbt.rekibun.or.jp/events/civic-canvas01-screening>



- 一人好きでちまちまやってたつもりが想像もしなかった未来に繋がった
- 今日のこのYAPCのLTもその1つ
- 何か1つ作るたびに「もっとこうしたい」が出てくる
- 一歩踏みだすと次につながる

みなさんも「好き」を詰め込んだ
自分だけの世界を創って
未来を創ってみませんか？

