

Generative Art with Processing

by @eduardlopez

@eduardlopez

- + TecnoUAB Cofounder
- + 2017 UAB CS Graduate
- + 2017 Badi.com as Data and Machine Learning Engineer
- + 2021 Wallapop.com as Lead Data Engineer
- +
+ linkedin.com/in/eduardlopezg
- + github.com/eduardlopez
- + twitter.com/eduardlopez
- + eduardlopez.com
- +

What's Generative Art?

- + Digital assets*
- + Created with a computer*
- + You provide the rules -> boundaries
- + The computer follows them

+

+

+

+

+

+

+

+

+

+

+

+

+

+

+

+

+

+

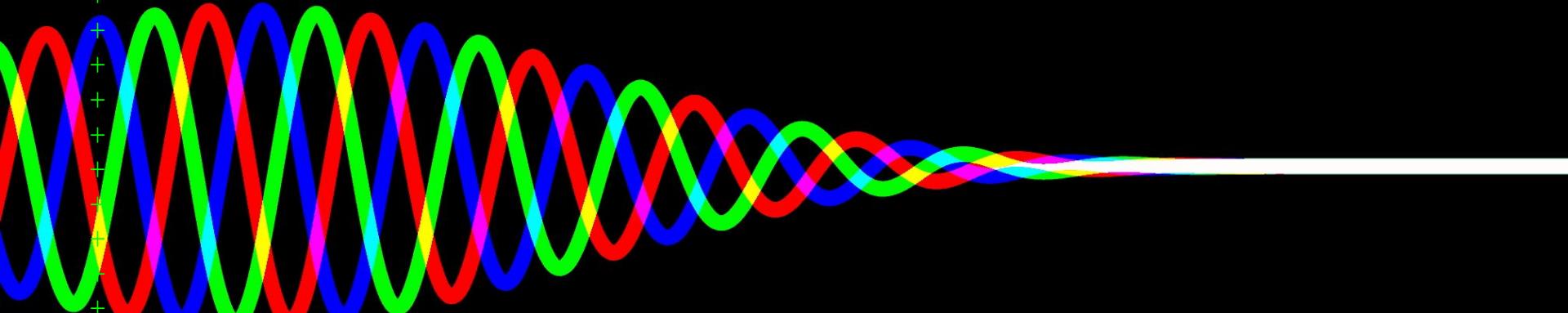
+

+

+

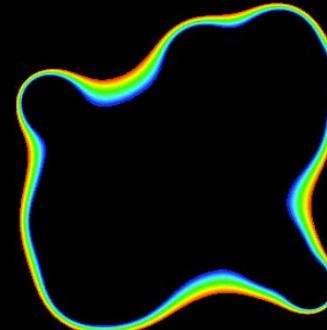
+

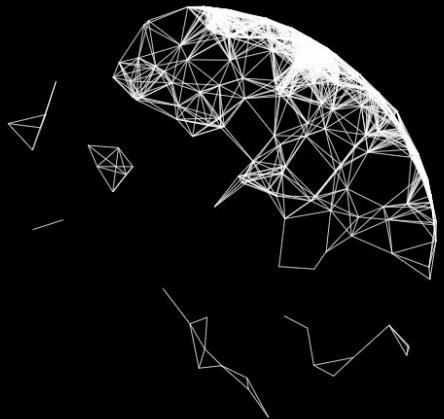
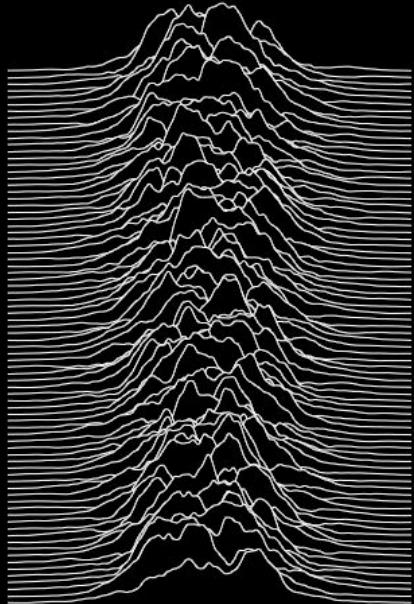
+



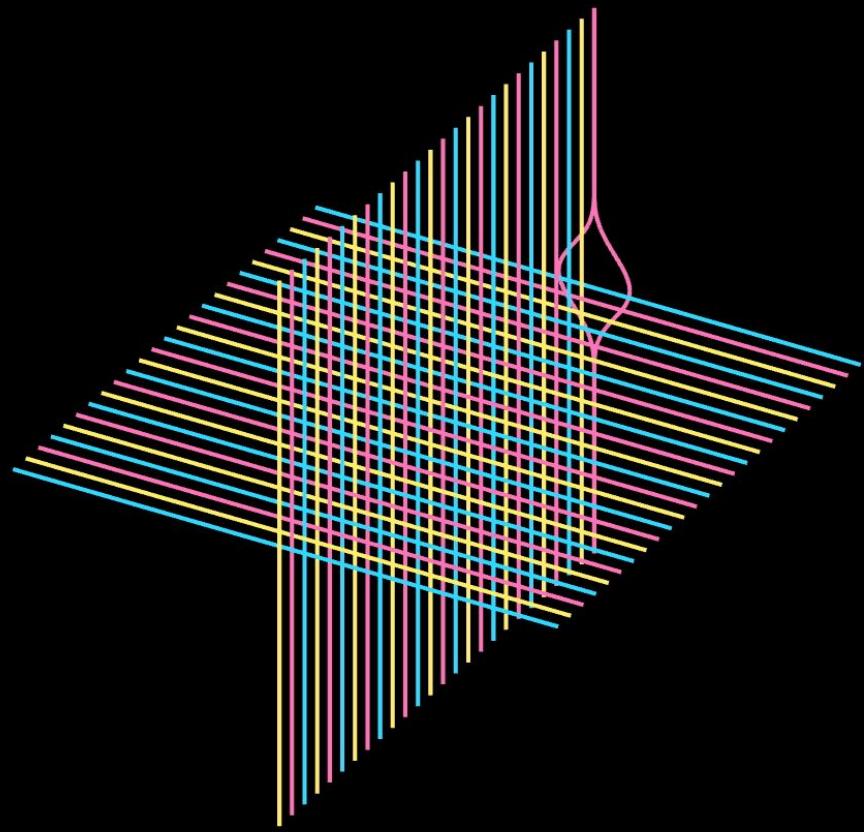
- + algorithmic art
- + computer-generated art
- + synthetic media

```
+ // Basile Pesin
+ // http://vertmo.github.io
+
+ // MetaBalls : p5.js implementation
+
+ var blobs = []
+
+ function setup() {
+   createCanvas(450, 450);
+   colorMode(HSB);
+   for (i = 0; i < 15; i++) blobs.push(new Blob(width/2,height/2));
+ }
+
+ function draw() {
+   background(0);
+   loadPixels();
+   for (x = 0; x < width; x++) {
+     for (y = 0; y < height; y++) {
+       let sum = 0;
+       for (i = 0; i < blobs.length; i++) {
+         let xdif = x - blobs[i].x;
+                     let ydif = y - blobs[i].y;
+         let d = sqrt((xdif * xdif) + (ydif * ydif));
+         sum += 10 * blobs[i].r*10 / d;
+       }
+       if(sum > 230 && sum <255) {
+         set(x, y, color(map(sum,230,255,0,255), 255, 255));
+       }
+     }
+   }
+   updatePixels();
+
+   for (i = 0; i < blobs.length; i++) {
+     blobs[i].update();
+     //blobs[i].show();
+   }
+ }
```



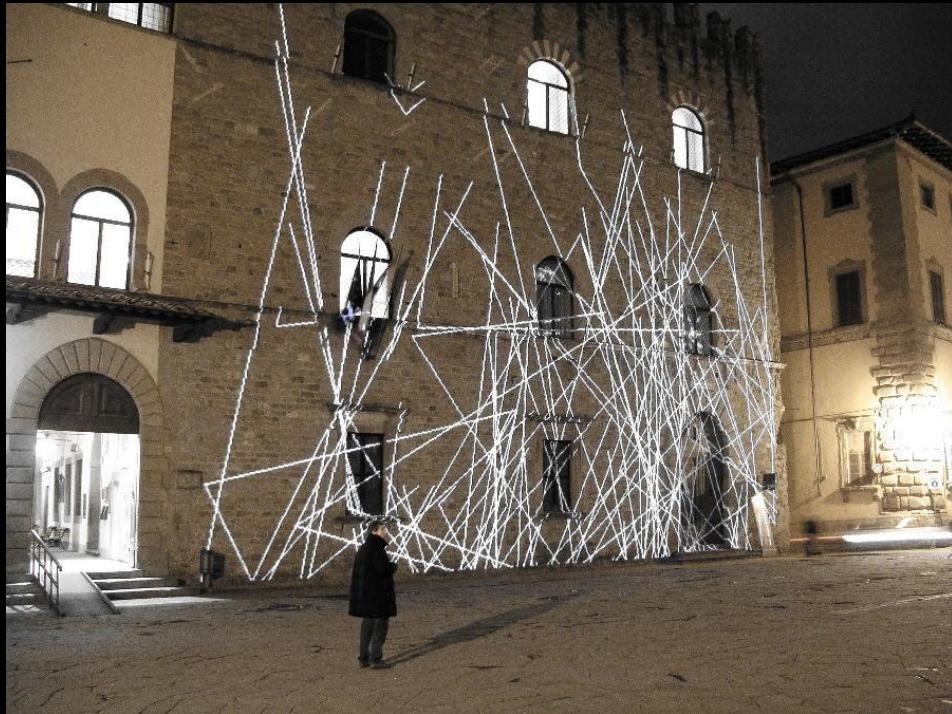
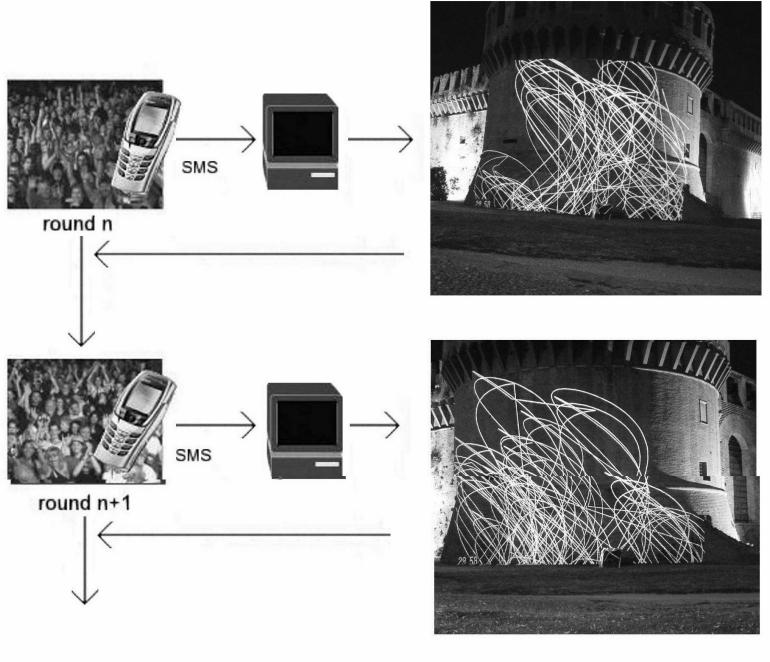


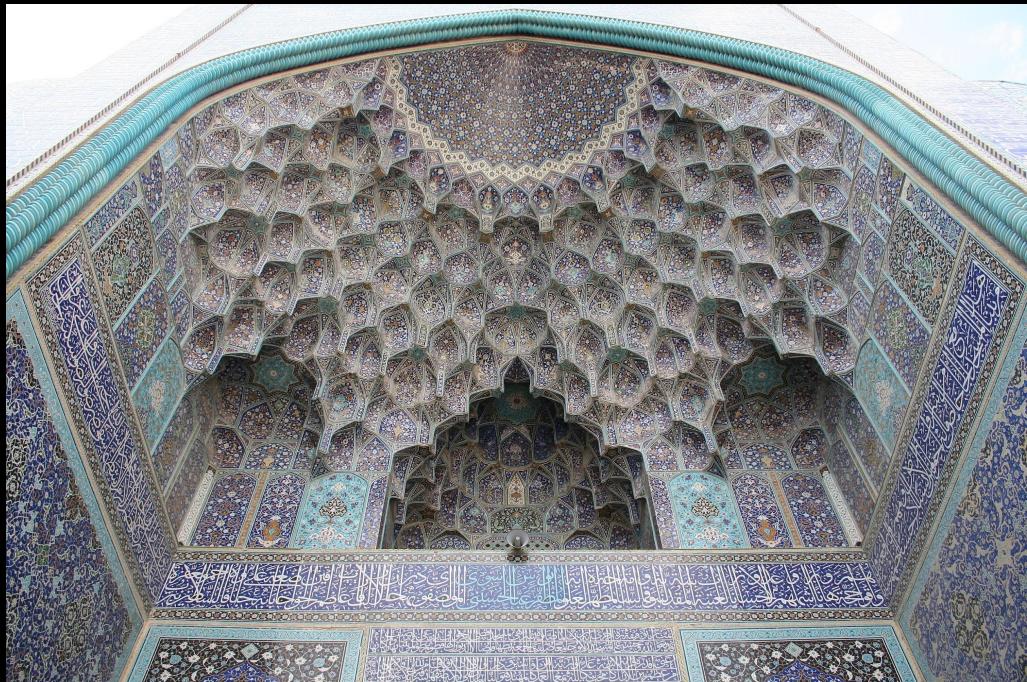
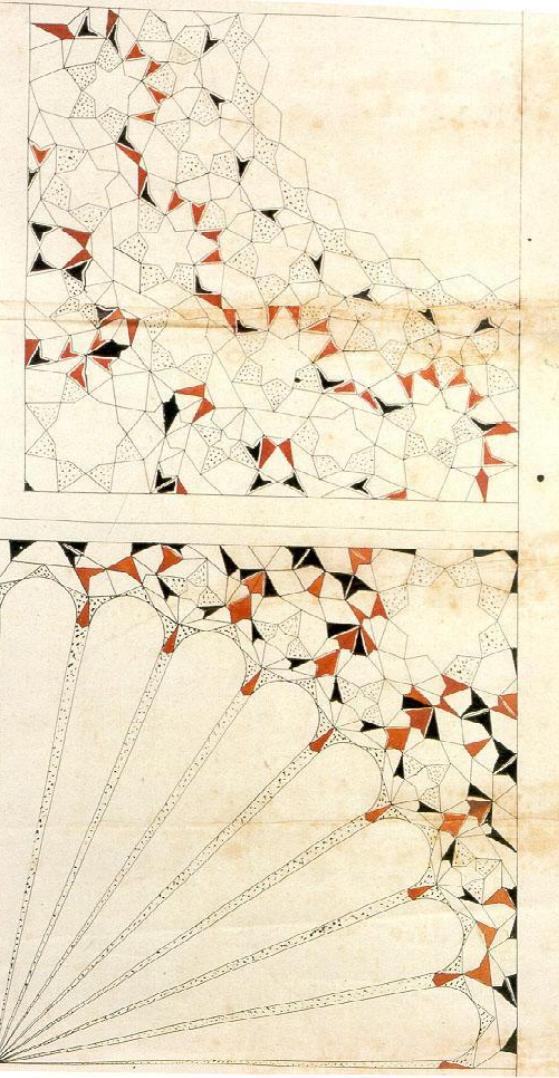
Sleeeeepy
Sleeeeepy
Sleeeeepy
Sleeeeepy
Sleeeeepy
Sleeeeepy
Sleeeeepy
Sleeeeepy

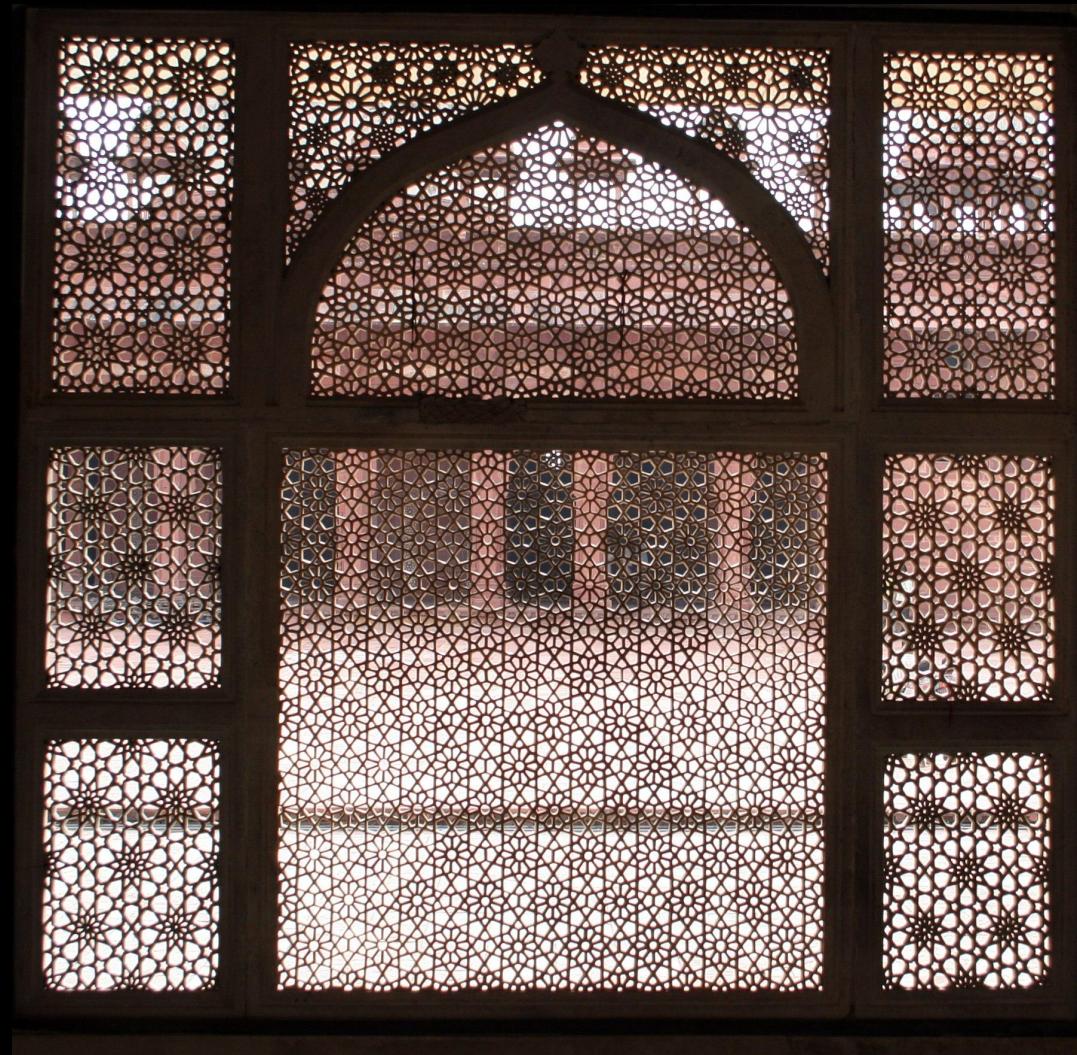




interactive + generative + public

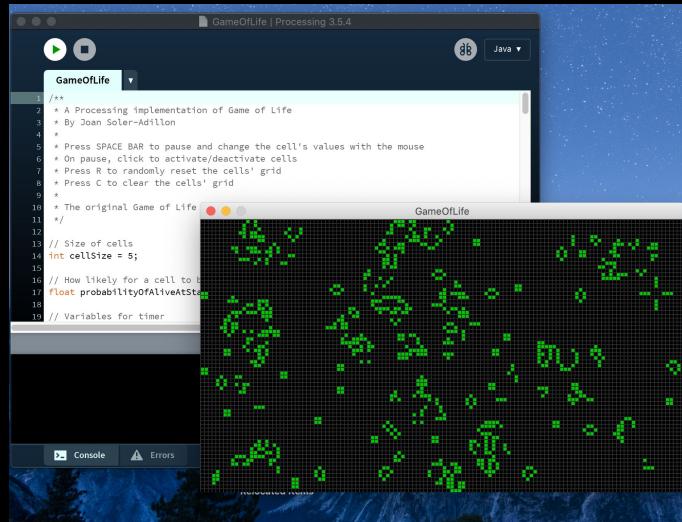






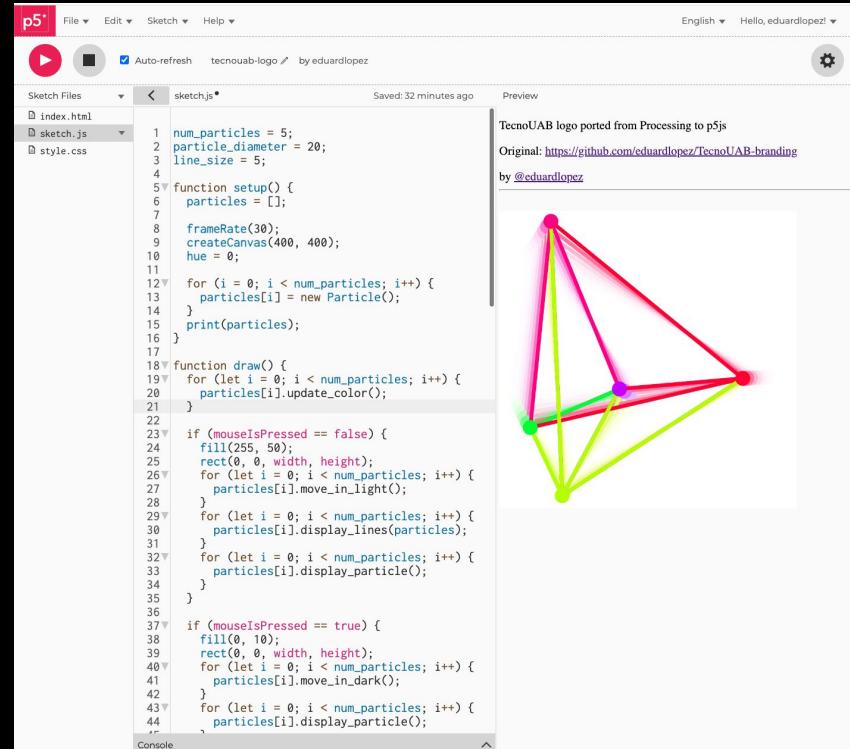
Tools | Processing.org

- + Graphical library + IDE
- + Based in Java with a layer of simplification
- + Self contained in an app, minimal friction to development
- + Main porting to javascript & python
- + Excellent learning materials: processing.org/tutorials/



Tools | p5js.org

- + The best porting of Processing to javascript
- + Web browser editor: <https://editor.p5js.org/>
- + Learn: p5js.org/get-started



The screenshot shows the p5.js web editor interface. On the left, the code editor displays a sketch named "sketch.js" with the following content:

```
p5.js
File ▾ Edit ▾ Sketch ▾ Help ▾
Auto-refresh tecnouab-logo* by eduardlopez
Sketch Files ▾ < sketch.js*
index.html
sketch.js
style.css
1 num_particles = 5;
2 particle_diameter = 20;
3 line_size = 5;
4
5 function setup() {
6   particles = [];
7
8   frameRate(30);
9   createCanvas(400, 400);
10  hue = 0;
11
12  for (i = 0; i < num_particles; i++) {
13    particles[i] = new Particle();
14  }
15  print(particles);
16
17
18 function draw() {
19  for (let i = 0; i < num_particles; i++) {
20    particles[i].update_color();
21  }
22
23  if (mouseIsPressed == false) {
24    fill(255, 50);
25    rect(0, 0, width, height);
26    for (let i = 0; i < num_particles; i++) {
27      particles[i].move_in_light();
28    }
29    for (let i = 0; i < num_particles; i++) {
30      particles[i].display_lines(particles);
31    }
32    for (let i = 0; i < num_particles; i++) {
33      particles[i].display_particle();
34    }
35  }
36
37  if (mouseIsPressed == true) {
38    fill(0, 10);
39    rect(0, 0, width, height);
40    for (let i = 0; i < num_particles; i++) {
41      particles[i].move_in_dark();
42    }
43    for (let i = 0; i < num_particles; i++) {
44      particles[i].display_particle();
45    }
}
}

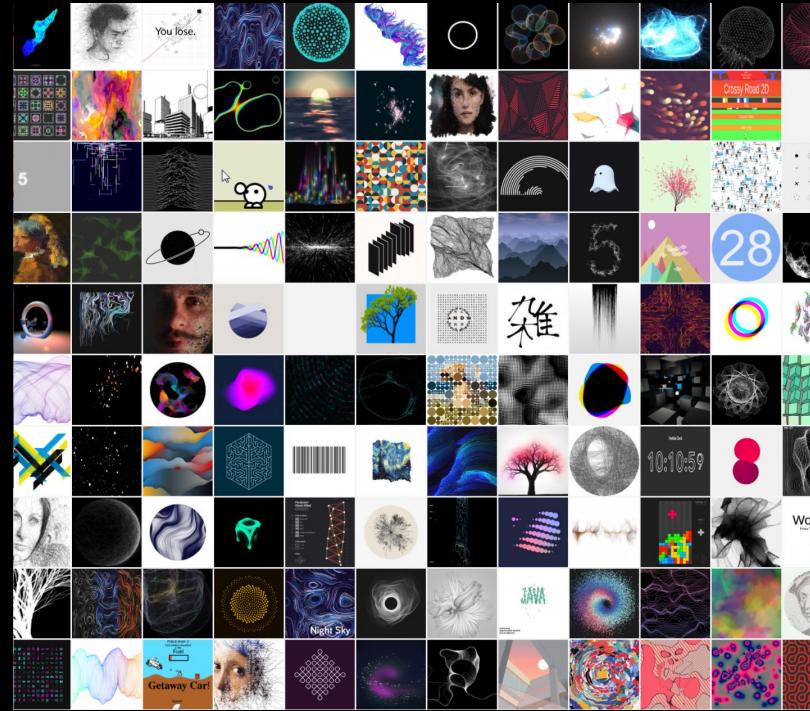
TecnoUAB logo ported from Processing to p5js
Original: https://github.com/eduardlopez/TecnoUAB-branding
by @eduardlopez
```

On the right, the preview window shows a 3D-like geometric logo composed of red and green lines forming a triangular structure.

Resources | openprocessing.org

+ Inspiration, Sharing

The screenshot shows the openprocessing.org homepage. The main header features a large, abstract geometric pattern composed of various colored triangles (blue, yellow, red, orange). Below the header, there's a prominent call-to-action box with the text "Coding is Beautiful". This box includes links for "Join" and "Sign in". To the right of this box, there's a "Create a Sketch" button. The page is divided into several sections: "Connect and Inspire" (with a "Explore sketches" link), "No Hassle Coding" (with a "Create a sketch" link), and "Teach and Learn" (with a "Learn more" link). At the bottom, there's a section titled "Sketches that are ❤'d this week" showing a row of small thumbnail sketches.

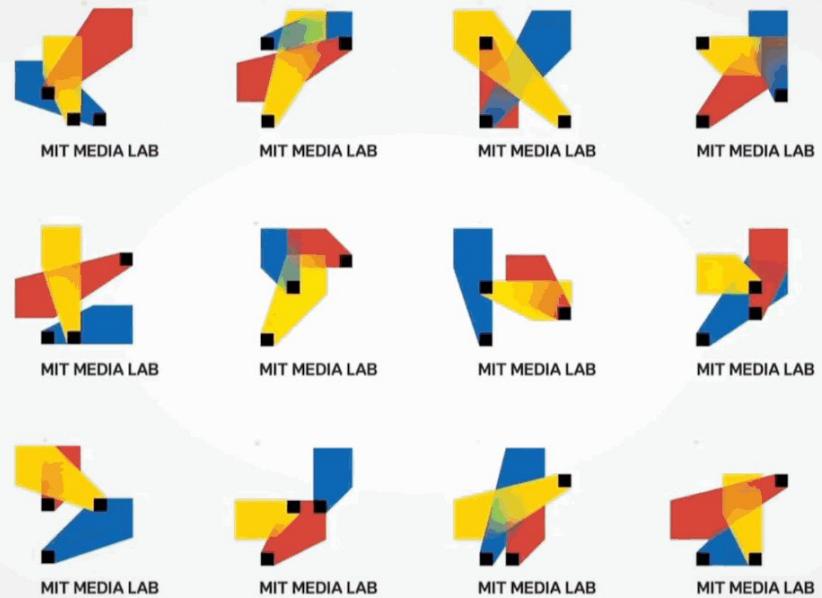


MIT Media Lab identity

+ 2011-2014



MIT MEDIA LAB



TecnoUAB Logo

- + Programmed with Processing
 - + <https://github.com/eduardlopez/TecnoUAB-branding>
- + Ported to js with p5.js
 - + <https://editor.p5js.org/eduardlopez/sketches/euHm0shLE>
- + Original meaning intent
 - + Particles: TecnoUAB members
 - + Lines: Ideas interchange done thanks the association
 - + Colors: Diversity of opinions
 - + if mouse click
 - + then dark -> no TecnoUAB -> no connections, no progress



Let's program!

- + Go to <https://editor.p5js.org/>
- + Copy the lines
- + Comment and uncomment each line to see different results
- + Let's comment what's happening



The screenshot shows the p5.js editor interface. At the top, there's a navigation bar with 'File', 'Edit', 'Sketch', and 'Help' menus, and language settings for 'English' and 'Hello, eduardlopez!'. Below the menu is a toolbar with play, stop, and refresh buttons, and an 'Auto-refresh' checkbox which is checked. The main workspace is divided into two sections: 'Sketch Files' on the left and the code editor on the right.

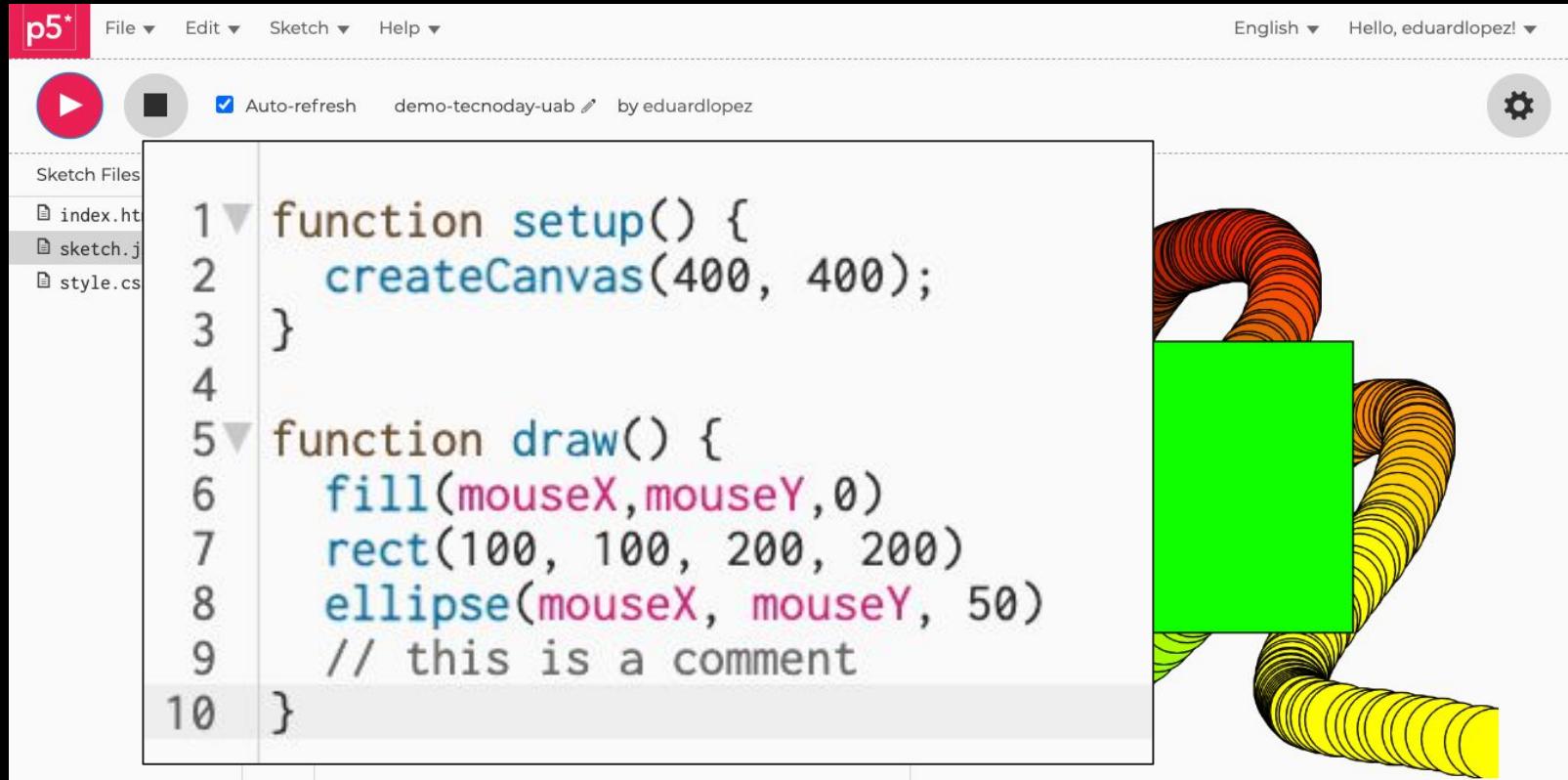
The 'Sketch Files' sidebar shows three files: 'index.html', 'sketch.js' (which is currently selected), and 'style.css'. The code editor contains the following P5.js code:

```
function setup() {
  createCanvas(400, 400);
}

function draw() {
  fill(mouseX, mouseY, 0)
  rect(100, 100, 200, 200)
  ellipse(mouseX, mouseY, 50)
  // this is a comment
}
```

To the right of the code editor is a preview window showing the output of the sketch. It features a large green square centered on a white background. Surrounding the square are several thick, colorful, spiral-like lines in shades of green, orange, red, and yellow, creating a sense of depth and motion.

- + Go to <https://editor.p5js.org/>
- + Copy the lines
- + Comment and uncomment each line to see different results
- + Let's comment what's happening



The screenshot shows the p5.js code editor interface. On the left, there's a sidebar titled "Sketch Files" with three files listed: "index.html", "sketch.js", and "style.css". The main workspace contains the following p5.js code:

```
1 function setup() {
2   createCanvas(400, 400);
3 }
4
5 function draw() {
6   fill(mouseX, mouseY, 0)
7   rect(100, 100, 200, 200)
8   ellipse(mouseX, mouseY, 50)
9   // this is a comment
10 }
```

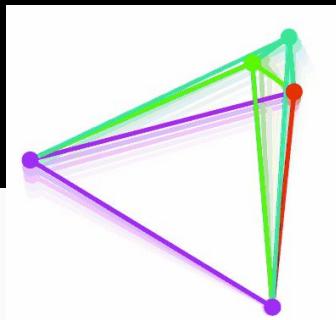
The output window on the right displays a white canvas with a green square centered at approximately [100, 100] and a yellow-orange snake-like coil at the bottom.

- + TecnoUAB p5.js logo
- + Interactive version:

<https://editor.p5js.org/eduardlopez/sketches/euHm0shLE>

```

1 num_particles = 5;
2 particle_diameter = 20;
3 line_size = 5;
4
5 function setup() {
6   particles = [];
7
8   frameRate(30);
9   createCanvas(400, 400);
10  hue = 0;
11
12  for (i = 0; i < num_particles; i++) {
13    particles[i] = new Particle();
14  }
15  print(particles);
16}
17
18 function draw() {
19  for (let i = 0; i < num_particles; i++) {
20    particles[i].update_color();
21  }
22
23  if (mouseIsPressed == false) {
24    fill(255, 50);
25    rect(0, 0, width, height);
26    for (let i = 0; i < num_particles; i++) {
27      particles[i].move_in_light();
28    }
29    for (let i = 0; i < num_particles; i++) {
30      particles[i].display_lines(particles);
31    }
32    for (let i = 0; i < num_particles; i++) {
33      particles[i].display_particle();
34    }
35  }
36
37  if (mouseIsPressed == true) {
38    fill(0, 10);
39    rect(0, 0, width, height);
40    for (let i = 0; i < num_particles; i++) {
41      particles[i].move_in_dark();
42    }
43    for (let i = 0; i < num_particles; i++) {
44      particles[i].display_particle();
45    }
46  }
47}
48
49 function mouseReleased() {
50  for (let i = 0; i < num_particles; i++) {
51    particles[i].init_dirs();
52  }
53}
54
55 class Particle {
56   constructor() {
57     this.diameter = particle_diameter;
58     this.line_size = line_size;
59
60     this.min_width = this.diameter;
61     this.max_width = width - this.diameter;
62
63     this.min_height = this.diameter;
64     this.max_height = height - this.diameter;
65
66     this.x = random(this.min_width, this.max_width);
67     this.y = random(this.min_height, this.max_height);
68     this.dir_x = 0;
69     this.dir_y = 0;
70     this.init_dirs();
71     this.speed = 1;
72     this.hue = random(0, 100);
73     this.color = color(this.hue, 100, 100);
74   }
75
76   init_dirs() {
77     this.dir_x = random(-10, 10);
78     this.dir_y = random(-10, 10);
79   }
80
81   update_color() {
82     if (this.hue > 100) {
83       this.hue = 0;
84     } else {
85       this.hue += 1;
86     }
87     colorMode(HSB, 100);
88     this.color = color(this.hue, 100, 100);
89   }
90}
91
92 move_in_light() {
93   if (this.x <= this.min_width) {
94     this.x = this.min_width + 1;
95     this.dir_x = this.dir_x * -1;
96   }
97   if (this.x >= this.max_width) {
98     this.x = this.max_width - 1;
99     this.dir_x = this.dir_x * -1;
100 }
101 if (this.y <= this.min_height) {
102   this.y = this.min_height + 1;
103   this.dir_y = this.dir_y * -1;
104 }
105 if (this.y >= this.max_height) {
106   this.y = this.max_height - 1;
107   this.dir_y = this.dir_y * -1;
108 }
109
110 this.x = this.x + this.dir_x;
111 this.y = this.y + this.dir_y;
112 }
113
114 move_in_dark() {
115   this.dir_x += random(2.0) - 1.0;
116   this.dir_x *= 0.96;
117   this.dir_y += random(2.0) - 1.0;
118   this.dir_y *= 0.96;
119   this.move_in_light();
120 }
121
122 display_lines(particles) {
123   strokeWeight(this.line_size);
124   stroke(this.color);
125   for (let i = 0; i < num_particles; i++) {
126     line(this.x, this.y, particles[i].x, particles[i].y);
127   }
128 }
129
130 display_particle() {
131   noStroke();
132   fill(this.hue, 100, 100);
133   ellipse(this.x, this.y, this.diameter, this.diameter);
134 }
135
136 }
```



Sources

<https://openprocessing.org/sketch/838276>
<https://vimeo.com/20488585>
<https://openprocessing.org/sketch/683686>
<https://openprocessing.org/sketch/434620>
<https://openprocessing.org/sketch/917861>
<https://openprocessing.org/sketch/733725>
<https://openprocessing.org/sketch/940954>
<https://openprocessing.org/sketch/941636>
<https://openprocessing.org/sketch/402961>
<https://aiartists.org/generative-art-design>
<https://www.youtube.com/watch?v=DZphIScEMtY>
<https://processing.org/>
<https://openprocessing.org/>
[https://en.wikipedia.org/wiki/Processing_\(programming_language\)](https://en.wikipedia.org/wiki/Processing_(programming_language))
https://en.wikipedia.org/wiki/Generative_art
<https://proyectoidis.org/maurizio-bolognini/>

