



PLAY WITH CODE: THE JOY OF P5.JS

PlayFest 2021



HELLO, I'M CAROL

- I love playing and creating with code.
- Ooh...cool. How did you make this?
- What happens if...
- I wonder if I can break it.
- People before code - always
- Learn, Build, Share - Openly



HELLO, I'M AMANDA

- I love creating and exploring the world with code.
- If possible, I choose to fix.
- Stop. Breathe. Examine.
- So many connections!
- So many possible spaces!
- Ephemeral >> permanent
- Expression >> expert

- Intros
- Why
- Connect to the Real World
- Play with p5.js
- Celebrate creations
- Share with friends and family

Processing p5.js Processing.py Processing for Android Processing for Pi

Processing Foundation

English Español 简体中文 한국어

p5.js

[Home](#) [Editor](#) [Download](#) [Donate](#) [Get Started](#) [Reference](#) [Libraries](#) [Learn](#) [Examples](#) [Books](#) [Community](#) [Showcase](#) [Forum](#) [GitHub](#) [Twitter](#)

Get Started

This page walks you through setting up a p5.js project and making your first sketch. The easiest way to start is using the [p5.js editor](#), you can open the web editor and can scroll down to [Your First Sketch](#). If you would like to work on the desktop version of p5.js you can scroll down to [downloading instructions](#).

Your First Sketch

In the [p5.js web editor](#) you should find the following code:

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

function draw() {
  background(220);
}
```

[Copy](#)

After `background(220);` include this line of code:
`ellipse(50, 50, 80, 80);`

Now your code should be like this:

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

function draw() {
  // ...
}
```

WHY P5.JS

Joy. Exploration. Confidence.

to them p5.js ^{but not} is...

Press space to interact with this visualization!



view the projects →



LOVE
BUN

MY
BEAR

DEAR
ME

MY MY

LOVE
BOT

BE MY
BEAR

TIME
BEAR

WINK
BEAR

BEAR
BEAR

LOOK
BEAR

BOOK

YOU
ARE
BARE

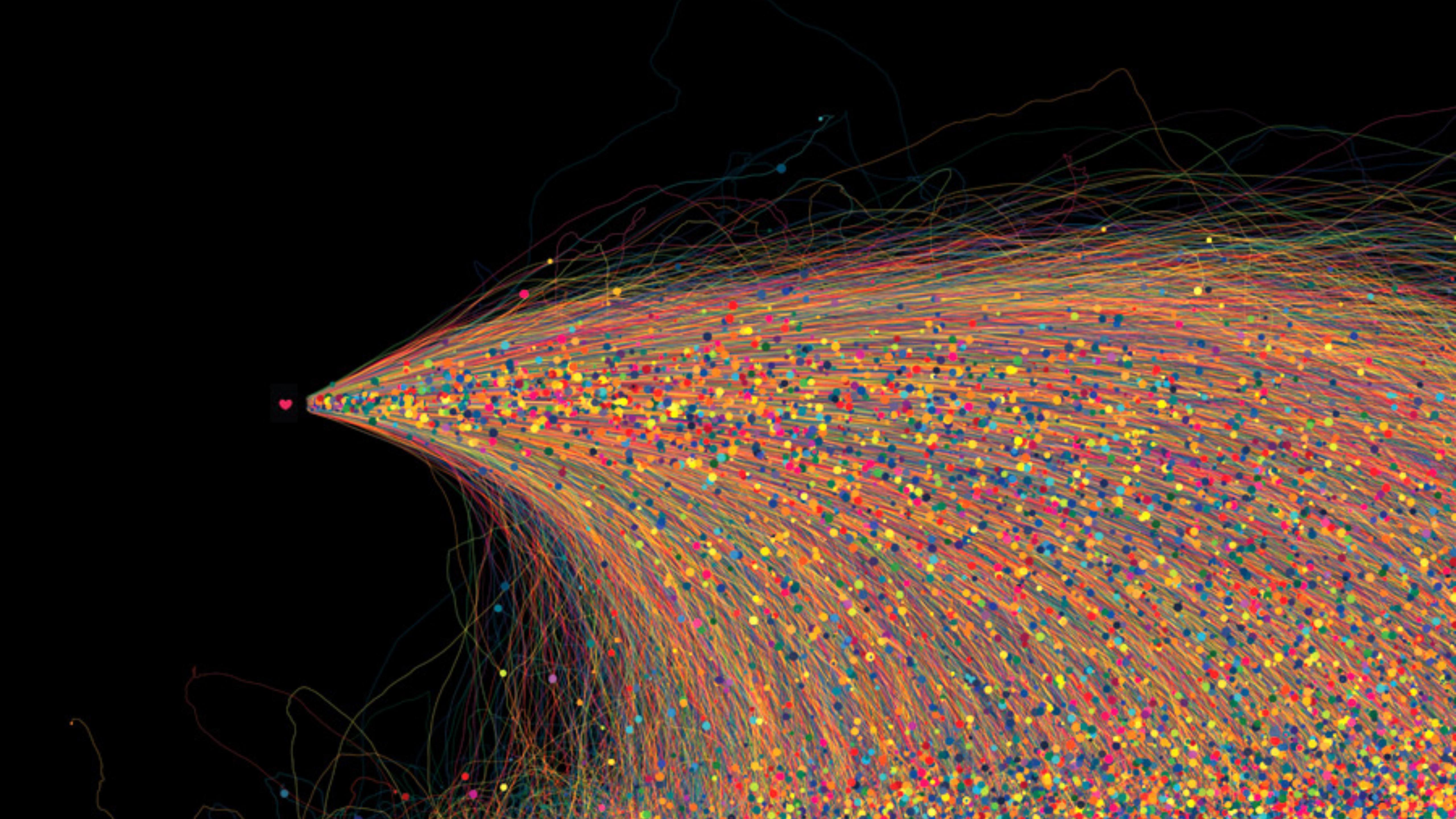
MAGE
LOVE

YAK
O
WAY

IN A
FAN











Joy Buolamwini - “Poet of Code”

p5.js 2020 Showcase

To learn more about the project and how it was made, click on the image of each card preview.
Feel free to leave feedback or a comment in our guestbook!

filter by: All Visual Data Visualization Game World
Text Sound Educational Teaching Simulation Tool
Camera

PLAY WITH P5.JS

Exploration and Possibilities

36 Days of Type x p5.js

Info

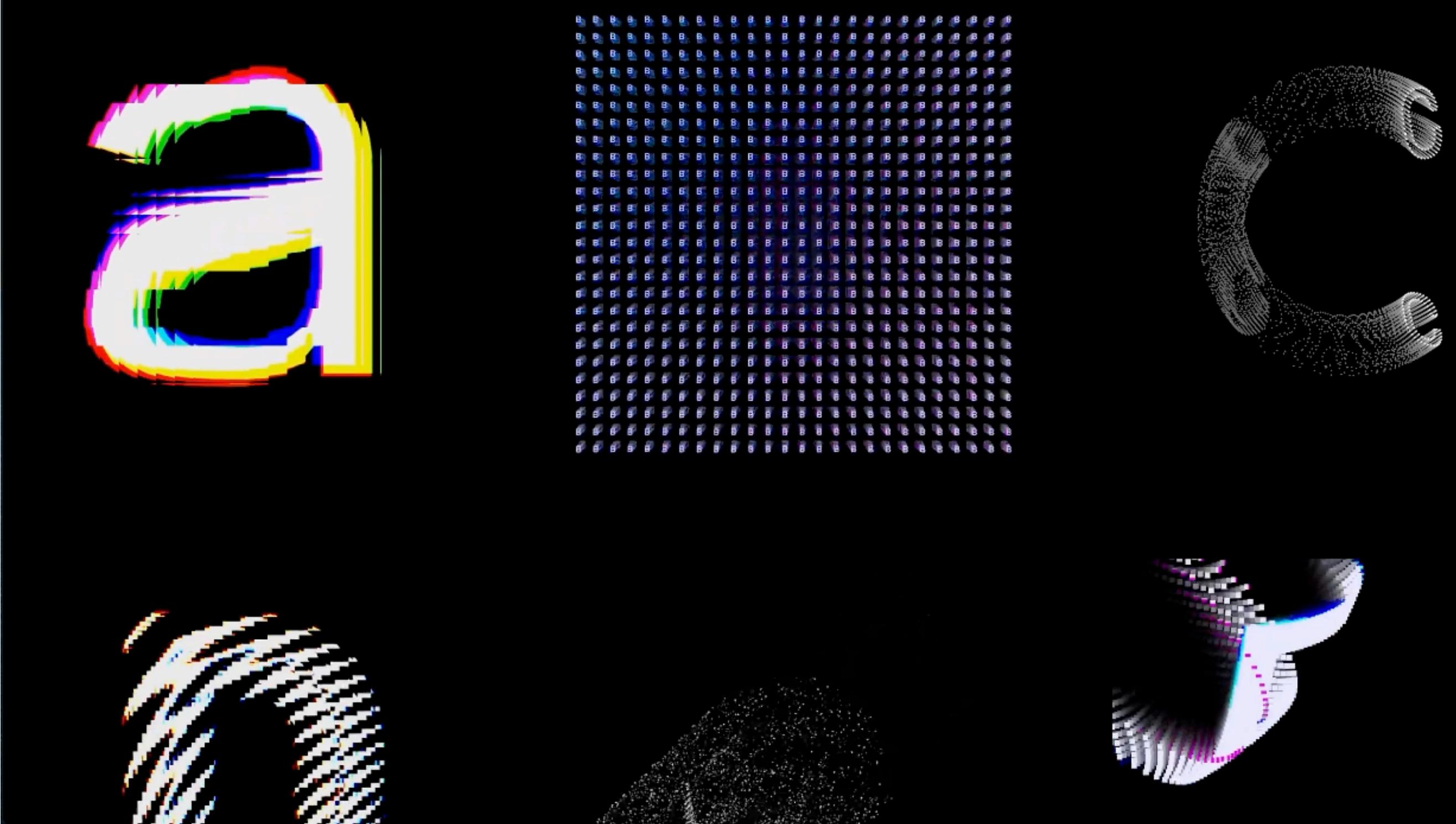
36 Days of Type is a yearly open challenge for designers, illustrators and graphic artists to interpret the 36 letters and numbers of the Latin alphabet over 36 consecutive days. For this year's edition I decided to code each character in p5.js, using the experience to learn more about the JavaScript library and creative coding.

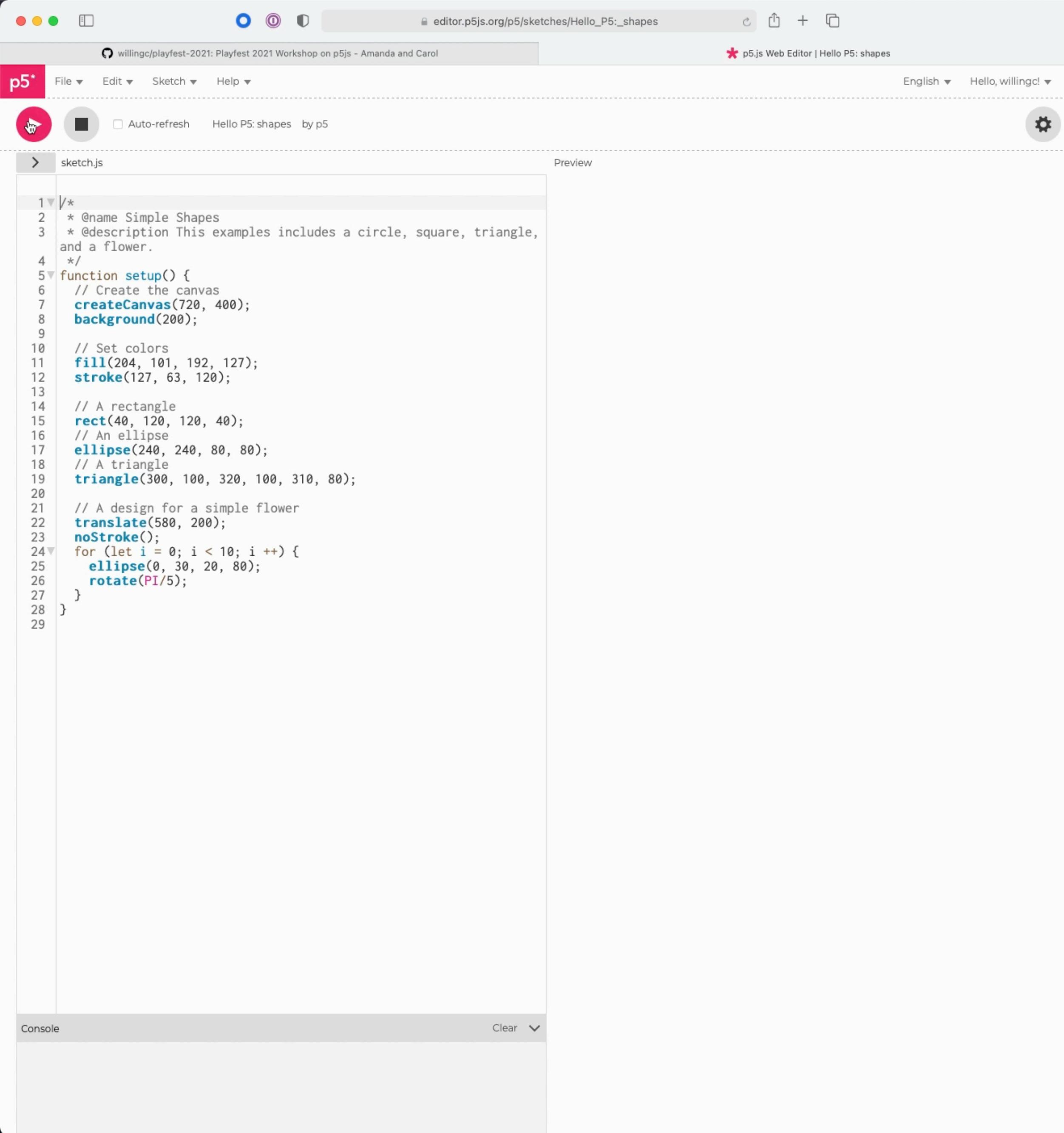
Shoutout to Superhi's Experimental Typography and Creative Coding courses, Dan Schiffman's aka The Coding Train youtube tutorials, Tim Rodenbroecker's Processing courses, and many many other artists for inspiration and support.

Links

[p5.editor ↗](#)

[Instagram ↗](#)





The screenshot shows the p5.js Web Editor interface. At the top, there are browser tabs for "editor.p5js.org/p5/sketches/Hello_P5:_shapes" and "p5.js Web Editor | Hello P5: shapes". The main window has a red header bar with the "p5*" logo, "File", "Edit", "Sketch", and "Help" menus. Below the header is a toolbar with icons for "Auto-refresh" and "Hello P5: shapes by p5". The central area contains a code editor with a file named "sketch.js" and a preview canvas below it. The code editor shows the following P5.js code:

```
1  /*
2   * @name Simple Shapes
3   * @description This examples includes a circle, square, triangle,
4   * and a flower.
5   */
6  function setup() {
7    // Create the canvas
8    createCanvas(720, 400);
9    background(200);
10
11   // Set colors
12   fill(204, 101, 192, 127);
13   stroke(127, 63, 120);
14
15   // A rectangle
16   rect(40, 120, 120, 40);
17   // An ellipse
18   ellipse(240, 240, 80, 80);
19   // A triangle
20   triangle(300, 100, 320, 100, 310, 80);
21
22   // A design for a simple flower
23   translate(580, 200);
24   noStroke();
25   for (let i = 0; i < 10; i++) {
26     ellipse(0, 30, 20, 80);
27     rotate(PI/5);
28   }
29 }
```

At the bottom of the editor, there is a "Console" tab with a "Clear" button.

P5.JS

-
- No installation required. Web Based IDE
- Lots of examples. Try it, change it, explore.
- Great documentation. Translations. Accessible.
- Appeals to the Senses
- Choose your own adventure. Empower the learner.
- Share it with friends

editor.p5js.org

The screenshot shows the p5.js code editor interface. The title bar reads "editor.p5js.org/p5/sketches/Input:_Clock". The menu bar includes "File", "Edit", "Sketch", and "Help". The toolbar has icons for play, stop, and refresh, with "Auto-refresh" checked. The sketch name is "Input: Clock by p5". The code editor window contains "sketch.js" with the following code:

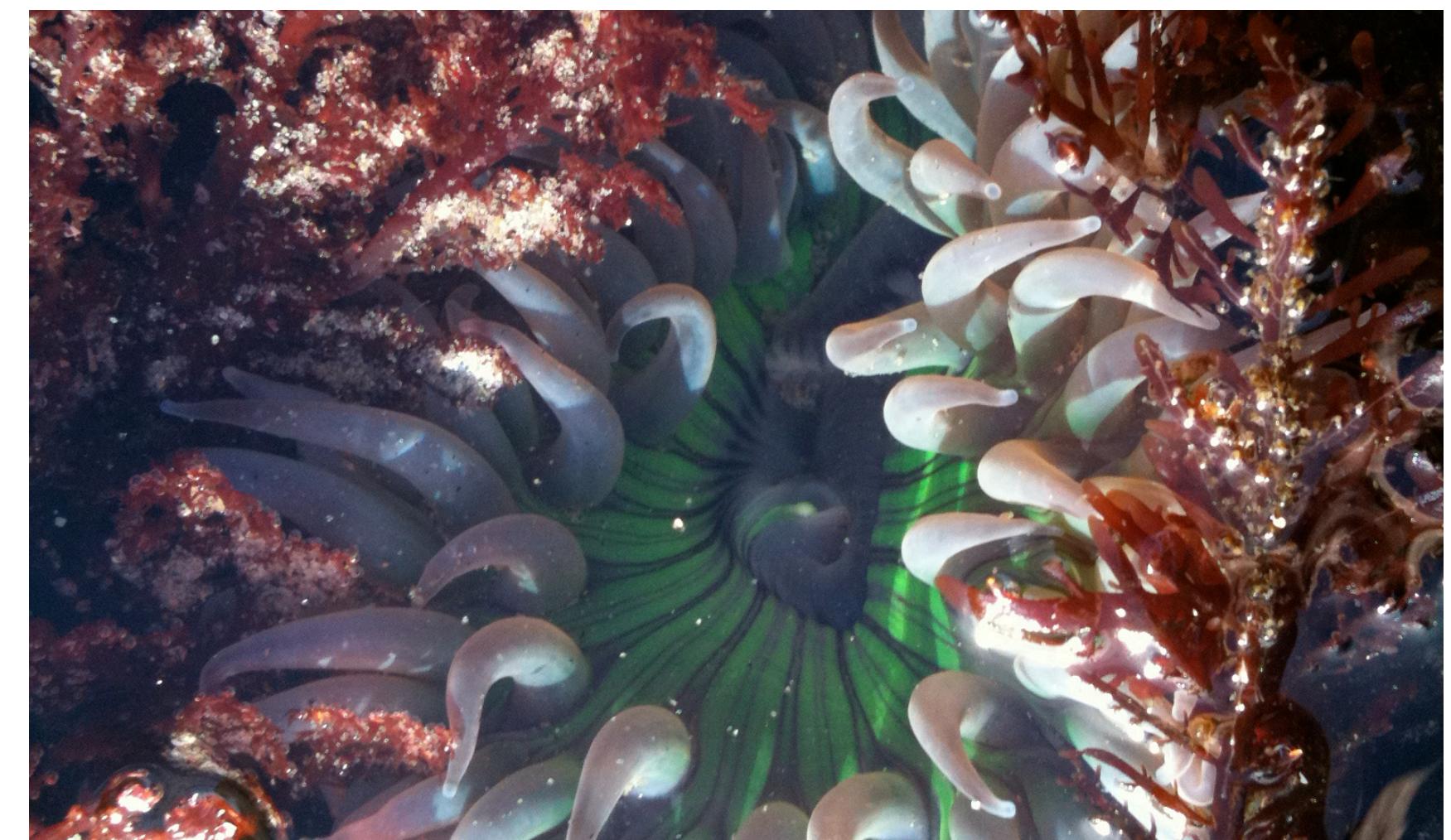
```
1  /*  
2   * @name Clock  
3   * @description The current time can be read with the second(),  
4   * minute(), and hour() functions. In this example, sin() and  
5   * cos() values are used to set the position of the hands.  
6   */  
7  let cx, cy;  
8  let secondsRadius;  
9  let minutesRadius;  
10 let hoursRadius;  
11 let clockDiameter;  
12  
13 function setup() {  
14   createCanvas(720, 400);  
15   stroke(255);  
16  
17   let radius = min(width, height) / 2;  
18   secondsRadius = radius * 0.71;  
19   minutesRadius = radius * 0.6;  
20   hoursRadius = radius * 0.5;  
21   clockDiameter = radius * 1.7;  
22  
23   cx = width / 2;  
24   cy = height / 2;  
25 }  
26  
27 function draw() {  
28   background(230);  
29  
30   // Draw the clock background  
31   noStroke();  
32   fill(244, 122, 158);  
33   ellipse(cx, cy, clockDiameter + 25, clockDiameter + 25);  
34   fill(237, 34, 93);  
35   ellipse(cx, cy, clockDiameter, clockDiameter);  
36  
37   // Angles for sin() and cos() start at 3 o'clock;  
38   // subtract HALF_PI to make them start at the top  
39   let s = map(second(), 0, 60, 0, TWO_PI) - HALF_PI;  
40   let m = map(minute() + norm(second(), 0, 60), 0, 60, 0, TWO_PI - HALF_PI);  
41   let h = map(hour() + norm(minute(), 0, 60), 0, 24, 0, TWO_PI * 2) - HALF_PI;  
42  
43   // Draw the hands of the clock  
44   stroke(255);  
45   strokeWeight(1);  
46   line(cx, cy, cx + cos(s) * secondsRadius, cy + sin(s) * secondsRadius);  
}
```

The preview area is empty. The status bar at the bottom shows "Console" and "Clear".

YES, YOU CAN...

- Set up / prep
- Production
- Clean up
- Parties
- Plays
- Cooking
- Writing
- and more...

editor.p5js.org



Choice, confidence, engagement...PLAY



CODE! PROGRAMMING WITH P5.JS

This video series focuses on the fundamentals of computer programming (variables, conditionals, iteration, functions & objects) using JavaScript. In particular it leverages the p5.js creative computing environment which is oriented towards visual displays on desktops, laptops, tablets or smartphones. The series is designed for computer programming novices.

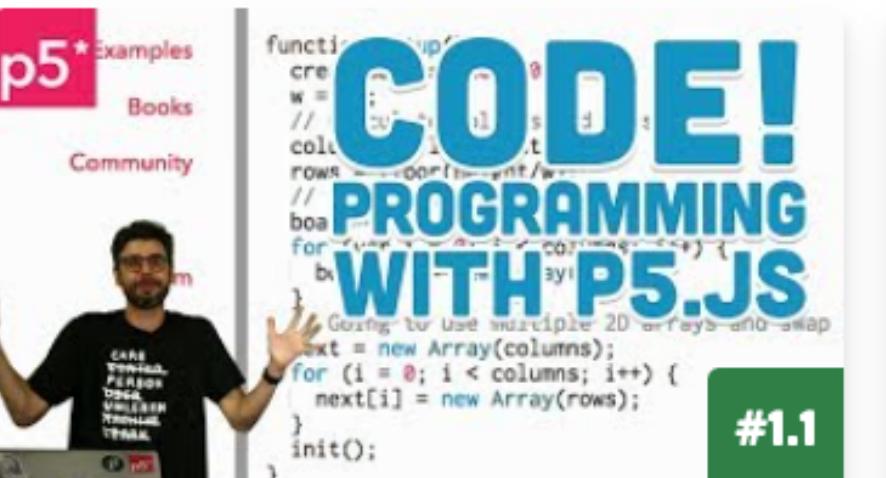
Hopefully if you make it to the end you'll be able to explain how computational media is different from traditional media, demonstrate an understanding of computer programming, and learn how to learn the tools they need to accomplish the projects that interest them in computational media.



TRAILER

12 Apr 2019

Code! Programming in JavaScript with p5.js for beginners.



PROGRAMMING FOR BEGINNERS WITH P5.JS

05 Sep 2018

In this video, I talk about the p5.js basics course, what I hope you will learn and the goals of the course.

[WATCH ON YT](#)

[TO THE LESSON](#)



P5.JS WEB EDITOR

06 Sep 2018

In this video, I cover getting set up with and using the p5.js Web Editor.

[WATCH ON YT](#)

[TO THE LESSON](#)

LET'S PLAY

- Interact
- Appealing
- Visual
- Sound
- Motion
- Individual - explore at your pace
- Excitement

editor.p5js.org
thecodingtrain.com

The screenshot shows the p5.js sketch editor interface. At the top, there are browser-style tabs and a title bar with the URL "editor.p5js.org/p5/sketches/Interaction:_kaleidoscope". Below the title bar, the p5 logo is visible, followed by menu items: File, Edit, Sketch, and Help. To the right of these are language settings (English) and a user greeting ("Hello, willingc!").

The main workspace is divided into two sections: "sketch.js" on the left and "Preview" on the right. The "sketch.js" section contains the JavaScript code for the "Interaction: kaleidoscope" sketch. The "Preview" section shows a blank canvas where the sketch will run.

sketch.js Code:

```
1  /*
2   * @name Kaleidoscope
3   * @description A kaleidoscope is an optical instrument with two
4   * or more reflecting surfaces tilted to each other in an angle. This
5   * example tries to replicate the behavior of a kaleidoscope. Set the
6   * number of reflections at the symmetry variable and start drawing
7   * on the screen. Adjust the brush size with the help of the slider.
8   * The clear screen as it says clears the screen. The save button
9   * will download a .jpg file of the art that you have created.
10 */
11 // Symmetry corresponding to the number of reflections. Change the
12 // number for different number of reflections
13 let symmetry = 6;
14
15 let angle = 360 / symmetry;
16 let saveButton, clearButton, mouseButton, keyboardButton;
17 let slider;
18
19 function setup() {
20   createCanvas(710, 710);
21   angleMode(DEGREES);
22   background(127);
23
24   // Creating the save button for the file
25   saveButton = createButton('save');
26   saveButton.mousePressed(saveFile);
27
28   // Creating the clear screen button
29   clearButton = createButton('clear');
30   clearButton.mousePressed(clearScreen);
31
32   // Creating the button for Full Screen
33   fullscreenButton = createButton('Full Screen');
34   fullscreenButton.mousePressed(screenFull);
35
36   // Setting up the slider for the thickness of the brush
37   brushSizeSlider = createButton('Brush Size Slider');
38   sizeSlider = createSlider(1, 32, 4, 0.1);
39
40   // Save File Function
41   function saveFile() {
42     save('design.jpg');
43   }
44
45   // Clear Screen function
46   function clearScreen() {
47     background(127);
48   }
49 }
```

The "Preview" section is currently empty, showing a blank white canvas.

Kaleidoscope

editor.p5js.org/p5/sketches/interaction:_kaleidoscope

 Auto-refresh

Interaction: Wavemaker by p5



> sketch.js

Preview

```
1  /*
2   * @name Wavemaker
3   * @description This illustrates how waves (like water waves) emerge
4   * from particles oscillating in place. Move your mouse to direct the
5   * wave.
6   * Contributed by Aatish Bhatia, inspired by <a
7   * href="https://beesandbombs.tumblr.com/post/45513650541/orbiters">Orbiters</a> by Dave Whyte.
8   */
9
10 let t = 0; // time variable
11
12 function setup() {
13   createCanvas(600, 600);
14   noStroke();
15   fill(40, 200, 40);
16 }
17
18 function draw() {
19   background(10, 10); // translucent background (creates trails)
20
21   for (let x = 0; x <= width; x = x + 30) {
22     for (let y = 0; y <= height; y = y + 30) {
23       // starting point of each circle depends on mouse position
24       const xAngle = map(mouseX, 0, width, -4 * PI, 4 * PI, true);
25       const yAngle = map(mouseY, 0, height, -4 * PI, 4 * PI, true);
26       // and also varies based on the particle's location
27       const angle = xAngle * (x / width) + yAngle * (y / height);
28
29       // each particle moves in a circle
30       const mvX = x + 20 * cos(2 * PI * t + angle);
```

Wavemaker

Console

editor.p5js.org/p5/sketches/interaction:_wavemaker

 Auto-refresh

3D: orbit control by p5



> sketch.js

Preview

```
1  /*
2   * @name Orbit Control
3   * @description Orbit control allows you to drag and move around the
4   * world.
5   */
6  function setup() {
7    createCanvas(710, 400, WEBGL);
8
9  function draw() {
10  background(250);
11  let radius = width * 1.5;
12
13 //drag to move the world.
14 orbitControl();
15
16 normalMaterial();
17 translate(0, 0, -600);
18 for (let i = 0; i <= 12; i++) {
19  for (let j = 0; j <= 12; j++) {
20    push();
21    let a = (j / 12) * PI;
22    let b = (i / 12) * PI;
23    translate(
24      sin(2 * a) * radius * sin(b),
25      (cos(b) * radius) / 2,
26      cos(2 * a) * radius * sin(b)
27    );
28  if (j % 2 === 0) {
29    cone(30, 30);
30  } else {
31    box(30, 30, 30);
```

Console

Orbit Control

https://editor.p5js.org/p5/sketches/3D:_orbit_control

 Auto-refresh

Simulate: penrose tiles by p5



> sketch.js •

Preview

```
1  /*
2   * @name Penrose Tiles
3   * @frame 710,400
4   * @description This is a port by David Blitz of the "Penrose Tile"
example from processing.org/examples
5  */
6
7 let ds;
8
9 function setup() {
10  createCanvas(710, 400);
11  ds = new PenroseLSystem();
12  //please, play around with the following line
13  ds.simulate(5);
14 }
15
16 function draw() {
17  background(0);
18  ds.render();
19 }
20
21 function PenroseLSystem() {
22  this.steps = 0;
23
//these are axiom and rules for the penrose rhombus l-system
24 //a reference would be cool, but I couldn't find a good one
25  this.axiom = "[X]++[X]++[X]++[X]++[X]";
26  this.ruleW = "YF++ZF----XF[-YF----WF]++";
27  this.ruleX = "+YF--ZF[--WF--XF]+";
28  this.ruleY = "-WF++XF[+++YF++ZF]-";
29  this.ruleZ = "--YF++++WF[+ZF++++XF]--XF";
30
31 }
```

Penrose Tiles

The background of the slide features a collage of various geometric patterns, likely generated by the p5.js library. These include intricate orange and brown line art on a white background, a light beige surface with small circular motifs, and a red patterned band. The overall aesthetic is minimalist and modern.

PLAY WITH P5.JS

editor.p5js.org

https://editor.p5js.org/willingc/collections/GI3_u3qH0

FOUNDATION MOTIVATION CONNECTION

p5.js opens a world of possibilities



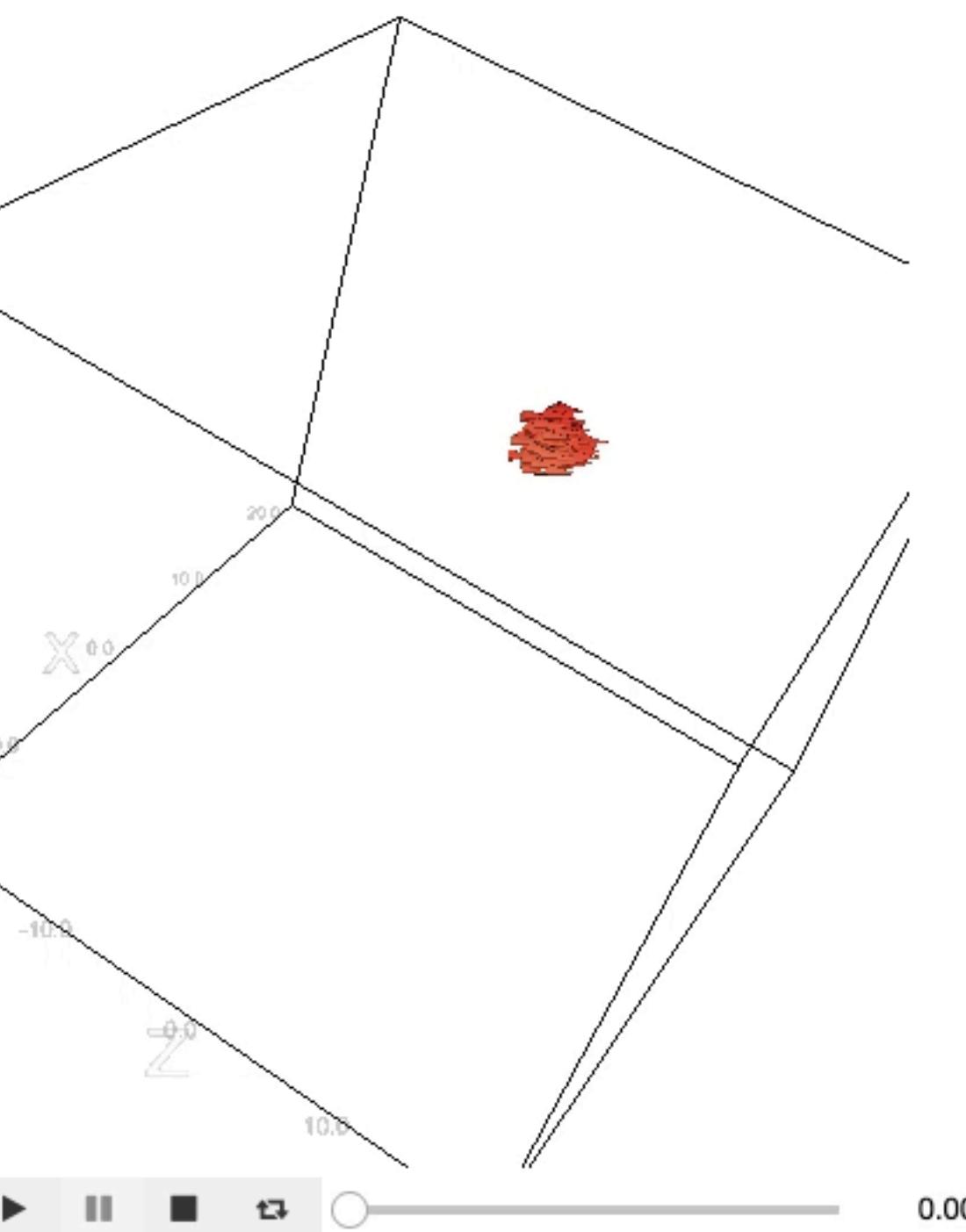
File Edit View Insert Cell Kernel Widgets Help

Trusted

Python 3



```
# use just rgba colors, not rgba
quiver = ipv.quiver(x, y, z, vx, vy, vz, size=5, color=colors[:,:,:,:3])
ipv.show()
```



```
In [93]: ipv.style.use('light')
```

```
In [94]: quiver.sequence_index = 3
```

```
In [96]: ipv.animation_control(quiver, interval=300)
```

```
In [97]: w = widgets.ToggleButtons(options=['arrow', 'sphere', 'cat'])
widgets.link((quiver, 'geo'), (w, 'value'))
w
```

arrow

sphere

cat

```
In [ ]:
```

jupyter.org

A screenshot of a GitHub repository page for `jtpio/p5-notebook`. The page shows the repository's code history, contributors, and other metadata.

Repository Summary:

- Code:** 16 issues, 2 pull requests, 97 commits
- Branches:** 4 branches
- Tags:** 0 tags
- Last Commit:** e2ce096 on Nov 15, 2020
- Contributors:** jtpio (1 commit)
- Languages:** TypeScript (88.7%), JavaScript (4.9%), Jupyter Notebook (3.7%), CSS (2.1%), HTML (0.6%)

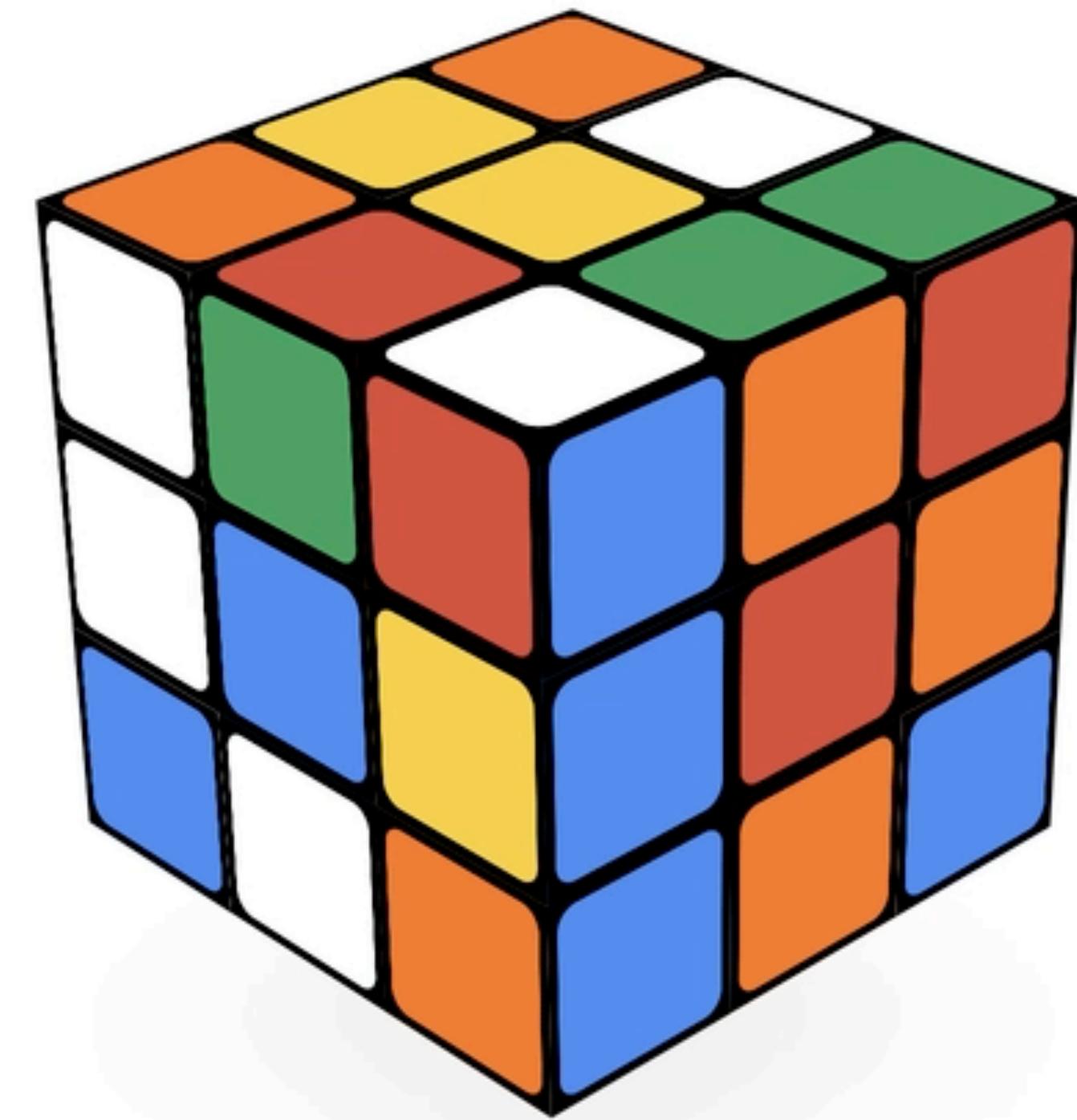
About: A minimal Jupyter Notebook UI for p5.js kernels running in the browser.

Contributors: jtpio (2)

Languages:

Language	Percentage
TypeScript	88.7%
JavaScript	4.9%
Jupyter Notebook	3.7%
CSS	2.1%
HTML	0.6%

p5-notebook.now.sh



google doodle



Make Music and Art Using Machine Learning

[Get Started](#)

[Try the Demos](#)

magenta.tensorflow.org

WHAT IS MAGENTA?

An open source research project exploring the role of machine learning as a tool in the creative process.

 Auto-refresh PitchDetection_Piano by ml5

> sketch.js

Preview

```
1 // Copyright (c) 2019 ml5
2 //
3 // This software is released under the MIT License.
4 // https://opensource.org/licenses/MIT
5
6 /* ===
7 ml5 Example
8 A piano using pitch Detection with CREPE
9 === */
10
11 // Pitch variables
12 let pitch;
13 let audioContext;
14 let audioStream;
15
16 // Keyboard variables
17 const cornerCoords = [10, 40];
18 const rectWidth = 90;
19 const rectHeight = 300;
20 const keyRatio = 0.58;
21 const scale = ['C', 'C#', 'D', 'D#', 'E', 'F', 'F#', 'G', 'G#', 'A',
22 'A#', 'B'];
23 let currentNote = '';
24 function setup() {
```

Console

Clear ▾

 Auto-refresh

Sentiment_Interactive by ml5



> sketch.js

Preview

```
1 let sentiment;
2 let statusEl;
3 let submitBtn;
4 let inputBox;
5 let sentimentResult;
6
7 function setup() {
8   noCanvas();
9   // initialize sentiment
10  sentiment = ml5.sentiment('movieReviews', modelReady);
11
12  // setup the html environment
13  statusEl = createP('Loading Model...');
14  inputBox = createInput('Today is the happiest day and is full of
rainbows!');
15  inputBox.attribute('size', '75');
16  submitBtn = createButton('submit');
17  sentimentResult = createP('sentiment score:');
18
19  // predicting the sentiment on mousePressed()
20  submitBtn.mousePressed(getSentiment);
21 }
22
23 function getSentiment() {
24   // get the values from the input
```

Console

Clear ▾

editor.p5js.org/ml5/sketches/XCFPKqlKrt



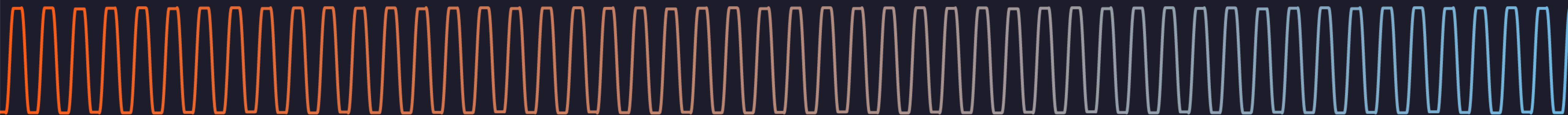
magic-sketchpad.glitch.me

NSYNTH: SOUND MAKER

Make unusual new sounds
with machine learning.

 PLAY

[How it works](#)



magenta.tensorflow.org/nsynth-instrument



magenta.tensorflow.org/nsynth-instrument

MARIMBA



TROMBONE



HARP

+

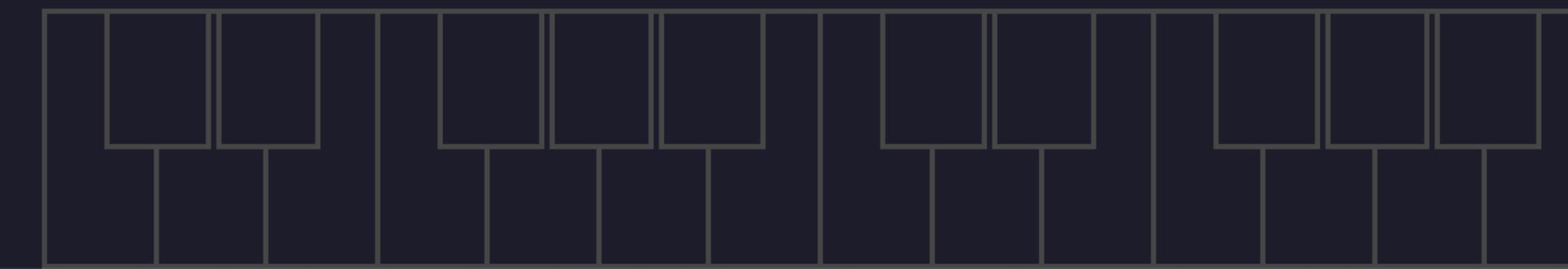
DOG



ELECTRIC
GUITAR



CLARINET



PLAY WITH MAGENTA

magenta.tensorflow.org

magic-sketchnpad.glitch.me

ml5js.org

editor.p5js.org/ml5/sketches



Thanks for sharing the joy of p5.js with us.