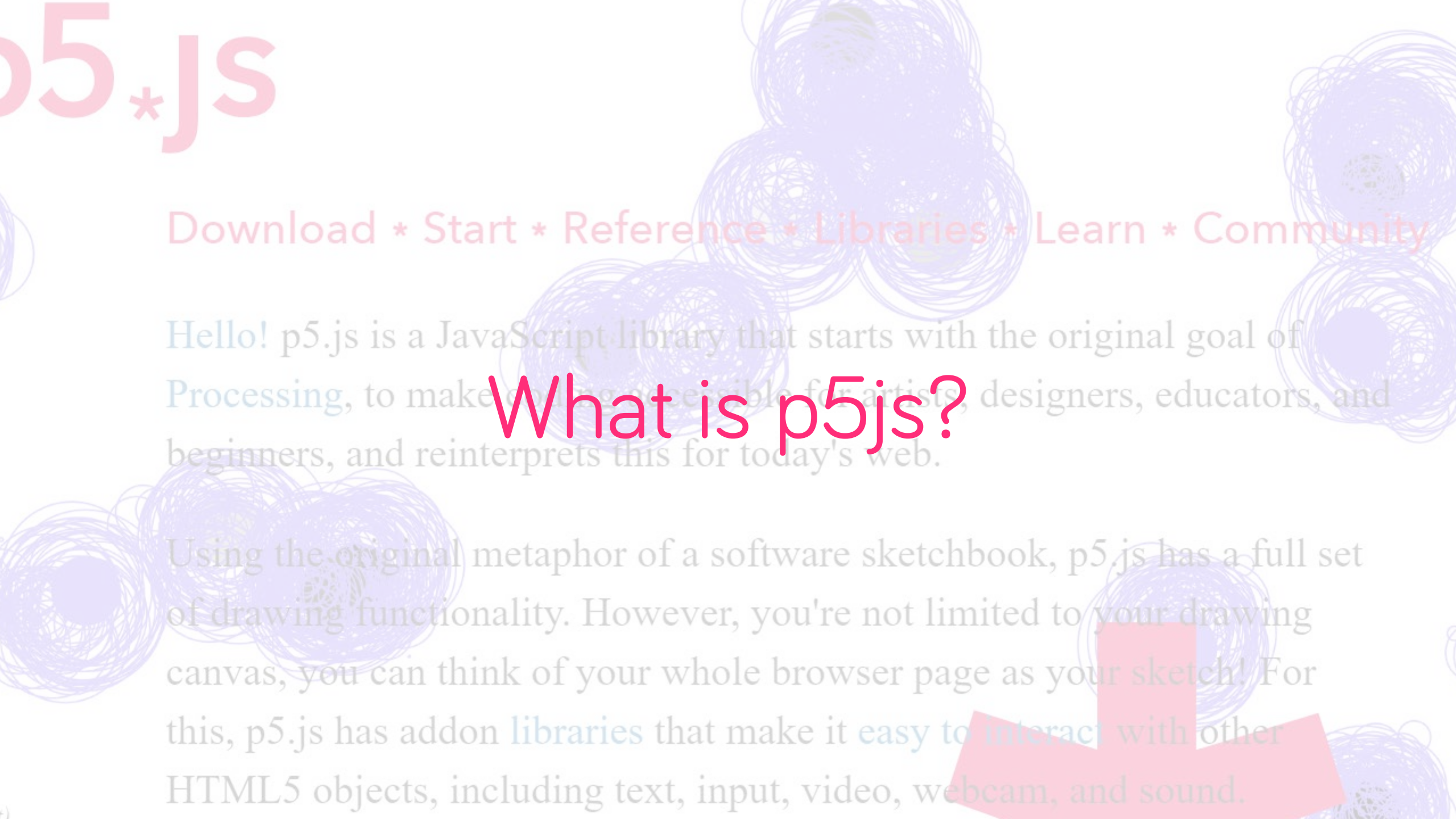
Introduction to p5js

Creative coding platform in JavaScript

What is p5js?


p5.js
Download * Start * Reference * Libraries * Learn * Community
Hello! p5.js is a JavaScript library that starts with the original goal of Processing, to make coding accessible for artists, designers, educators, and beginners, and reinterprets this for today's web.
Using the original metaphor of a software sketchbook, p5.js has a full set of drawing functionality. However, you're not limited to your drawing canvas, you can think of your whole browser page as your sketch! For this, p5.js has addon libraries that make it easy to interact with other HTML5 objects, including text, input, video, webcam, and sound.

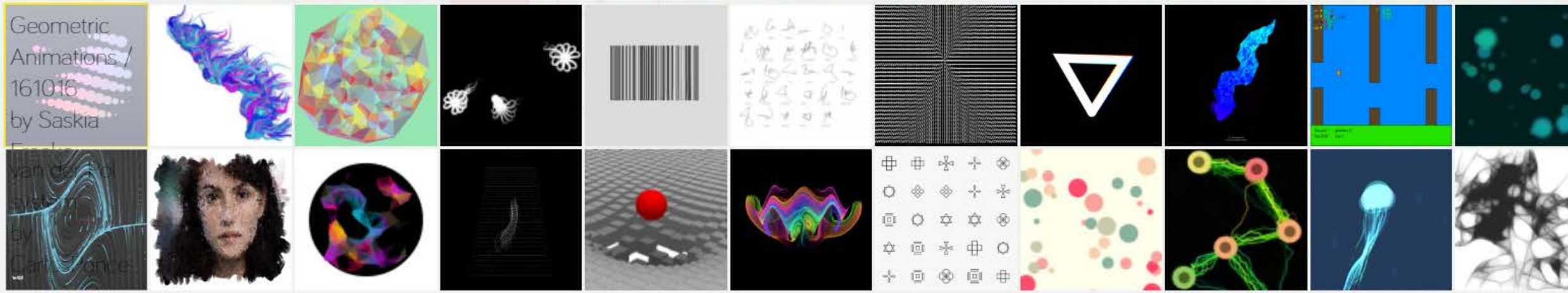
What is p5js?

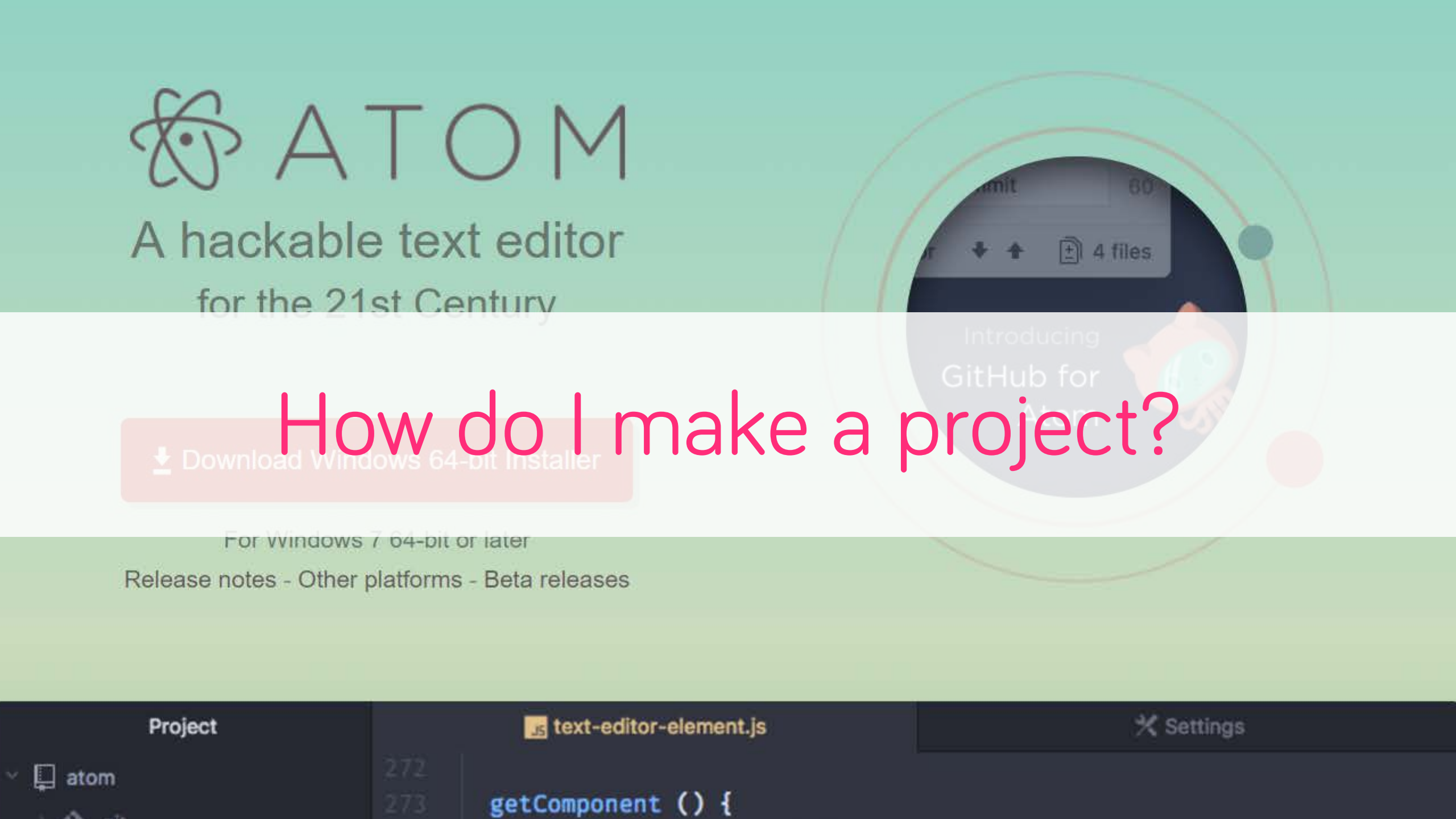
- p5 is a JavaScript library
- It can be combined with other JavaScript libraries carefully but it should control the canvas.
- JavaScript is a Object Oriented (OO) scripting language used client side to add functionality, graphics, sound and more to a webpage
- JavaScript is the wild west of languages. There are as many flavors of JavaScript as there are ice cream.

Algorithmic Design

for the Creative Hive

Openprocessing.org




ATOM
A hackable text editor
for the 21st Century
4 files
Introducing
GitHub for
Download Windows 64-bit Installer
For Windows 7 64-bit or later
Release notes - Other platforms - Beta releases
How do I make a project?
Project
text-editor-element.js
Settings
atom
272
273
getComponent () {

Foundation, a nonprofit organization devoted to advancing the role of programming within the visual arts through developing p5.js.

Donate here.

Complete Library

p5.js complete

*

Includes:

p5.js, p5.dom.js, p5.sound.js, and an example project

Version 0.5.4 (October 1, 2016)

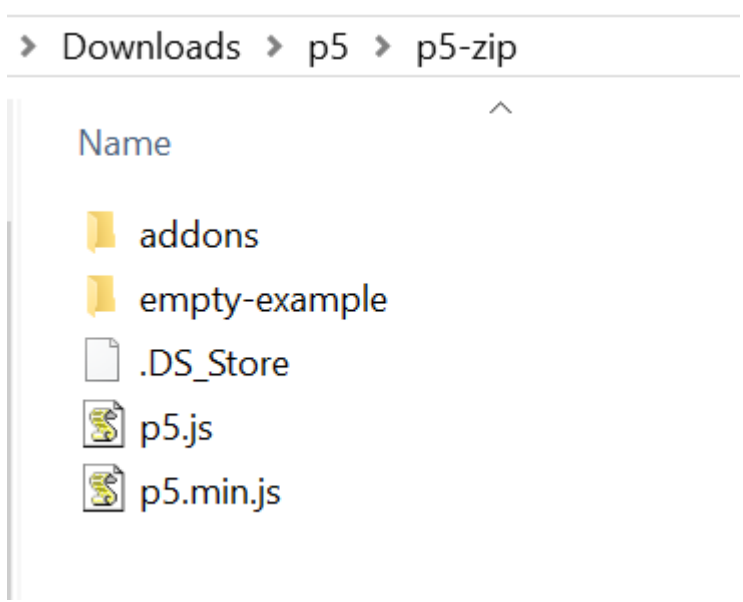
Download p5js complete from p5js.org

Single Files

p5.js

p5.min.js

CDN



Copy the empty-example folder and rename it. It can live anywhere on your hard drive.

Making a project

- Download p5
- Download atom, a software integrated developing environment (IDE)
- You can add packages to Atom to extend the functionality: browser-plus, p5jxs-autocomplete, p5js-snippets are 3 good ones to get started.
- Markdown is an easy way to write syntax. Consider adding some markdown functionality for fiction and documentation writing as an added win.

Where do I write my code?

```
Author: Phoenix Perry
Code Liberation
Created: 22-07-17
*/


var gameState = "startScreen"; //this
will change as we play the game
var isTouching = false; //checks if
circles are touching
var score = 0;
var speed = 5;
var ui_color;
var ui_text_color;
var ui_accent_color;
var enemySize = 100; //size of emeny
enemy_y_pos = 0; //for keeping track of
enemy locaiton

var playerSize = 100; //size of player
```

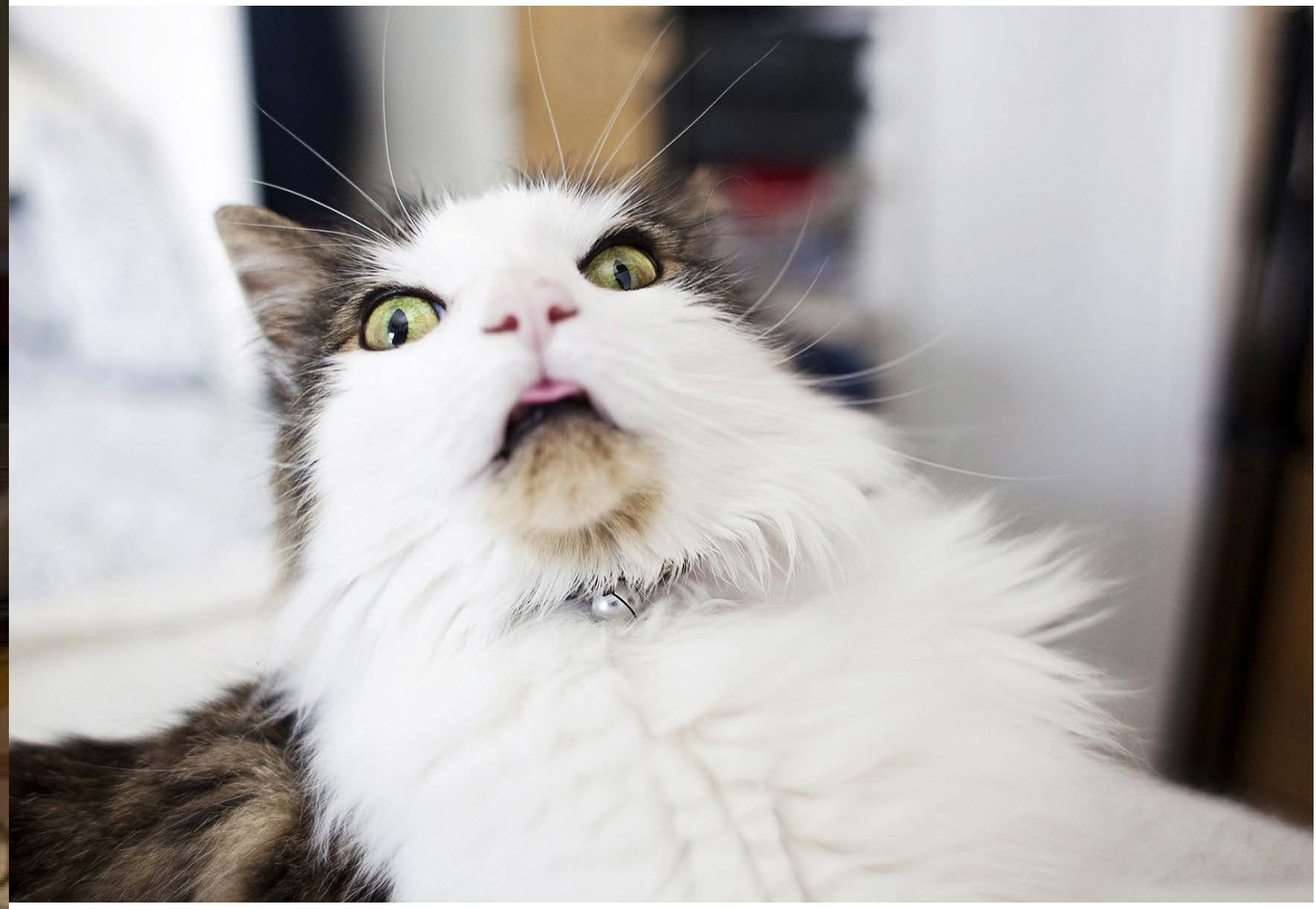
```
<head>
  <script
  src="https://cdnjs.cloudflare.com/aja
  p5.min.js"></script>
  <script
  src="https://cdnjs.cloudflare.com/aja
  addons/p5.dom.min.js"></script>
  <script
  src="https://cdnjs.cloudflare.com/aja
  addons/p5.sound.min.js"></script>
  <script src="sketch.js"></script>
  <link rel="stylesheet" type="text/css
  <style> body {padding: 0; margin: 0;} <
</head>
<body>
</body>
</html>
```

Can I take notes in my code?

Yes. Do it always

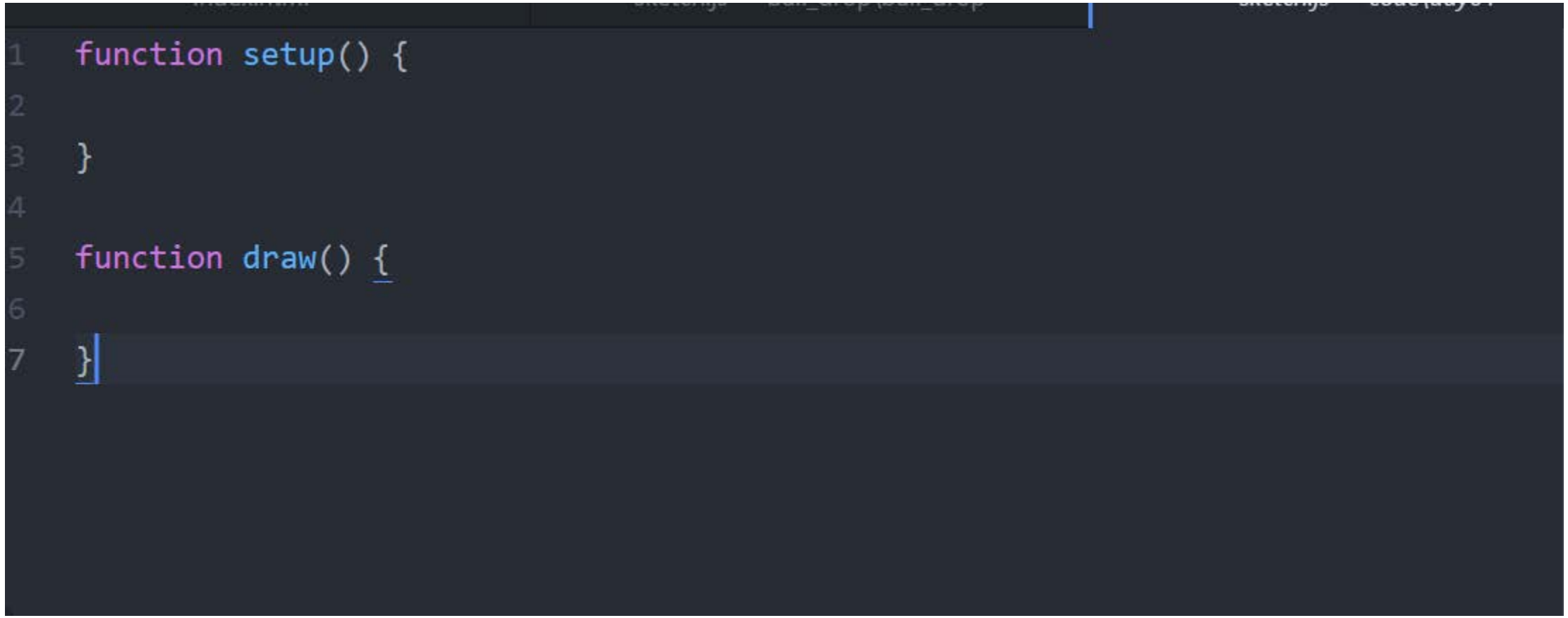
```
//for a one line comment.
/*for an entire novel that will
take up multiple lines about your
love for cats*/
```

Two weeks later

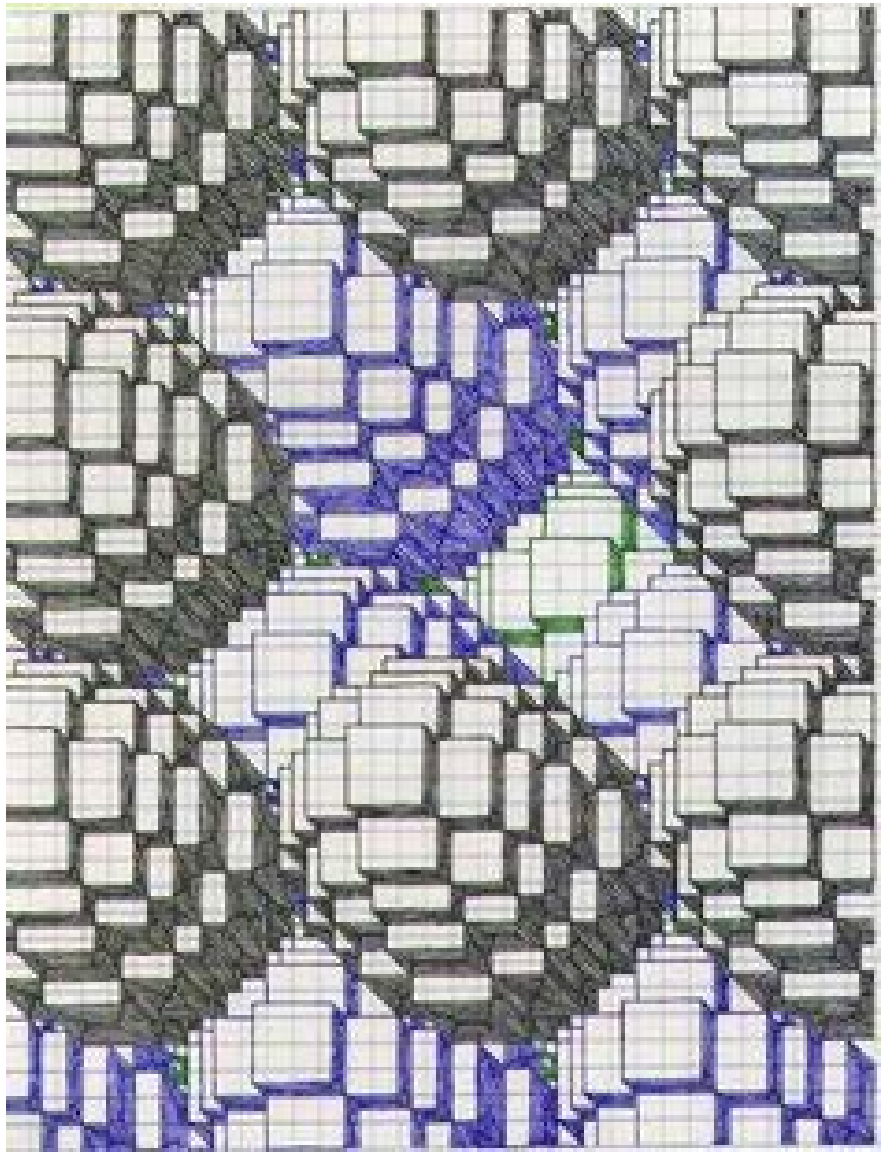


How do I make a project?

What is going on in sketch.js?

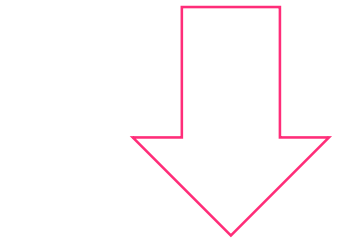


`setup()` is where you get everything prepared

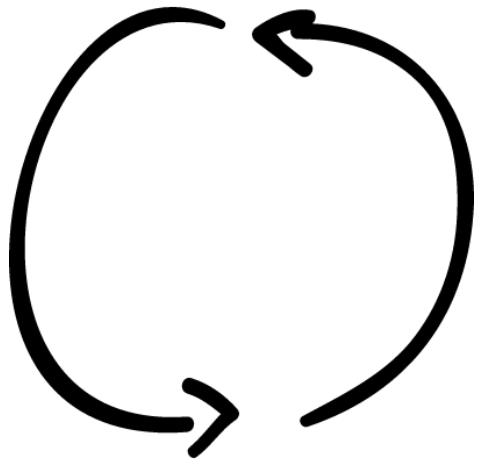


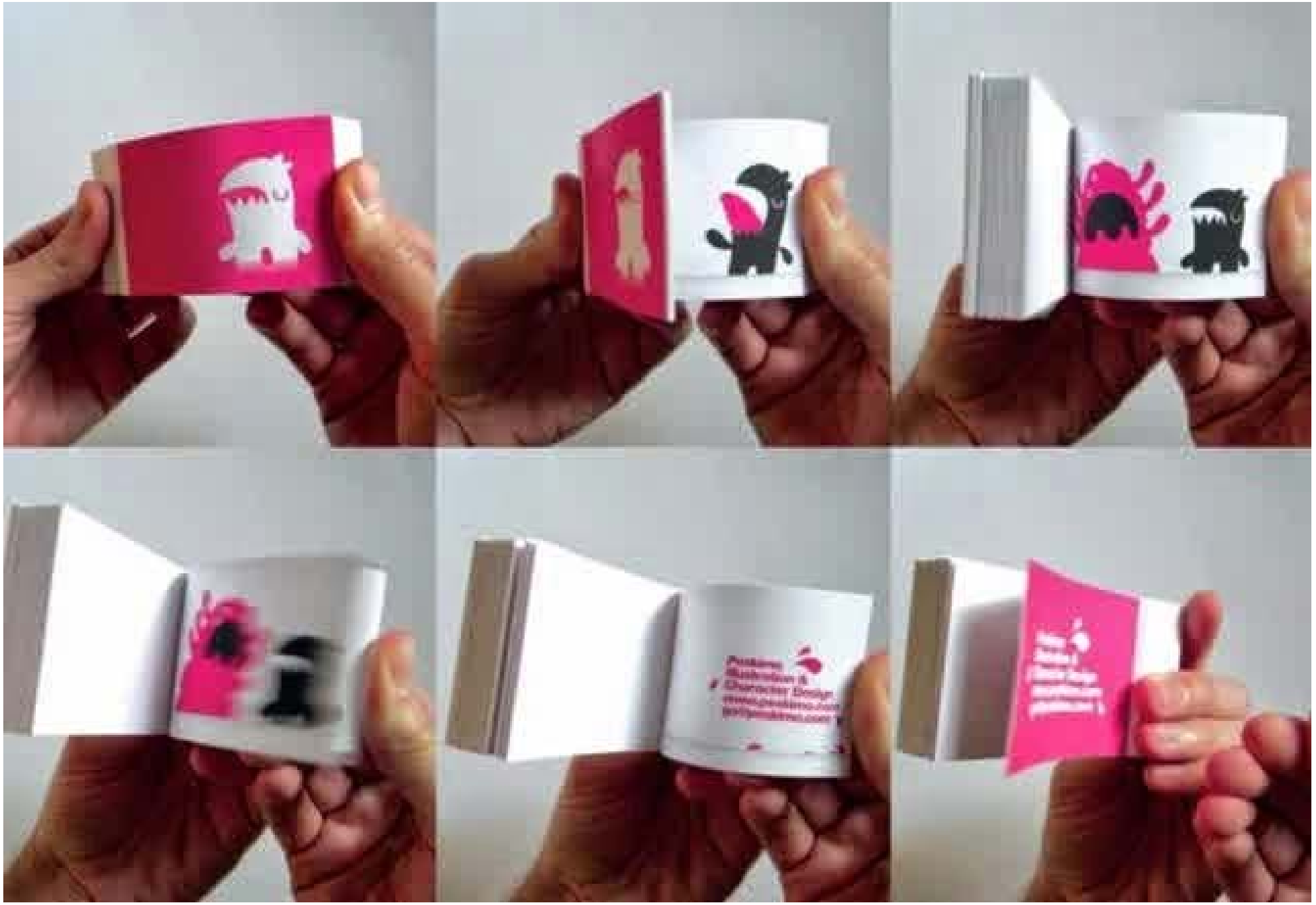
`draw()` is where you draw your art and update your mathy bits

setup()



draw()





Main functions of p5js

`setup()`: runs just once at the start of the app
Good place to set up initial player variables like

```
player_health=100;
```

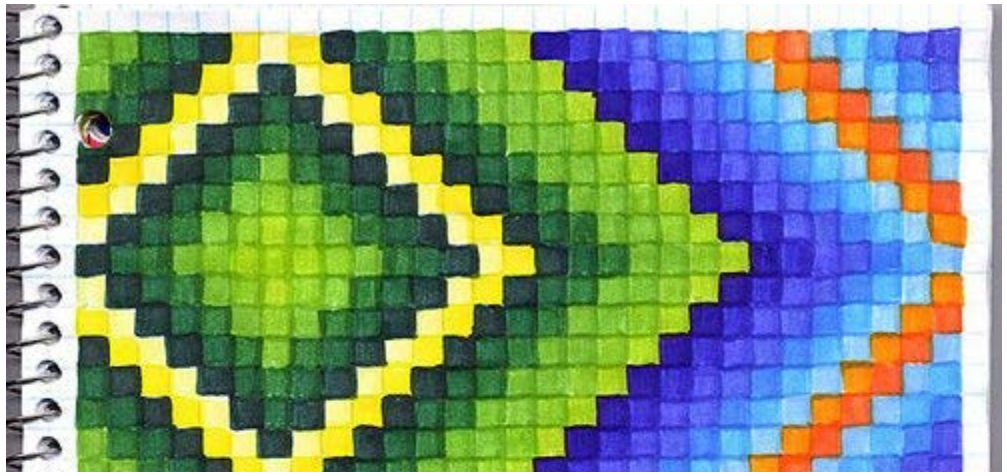
`draw()`: runs continuously once per frame.
It's where you should put your drawings like

```
player.draw();
```

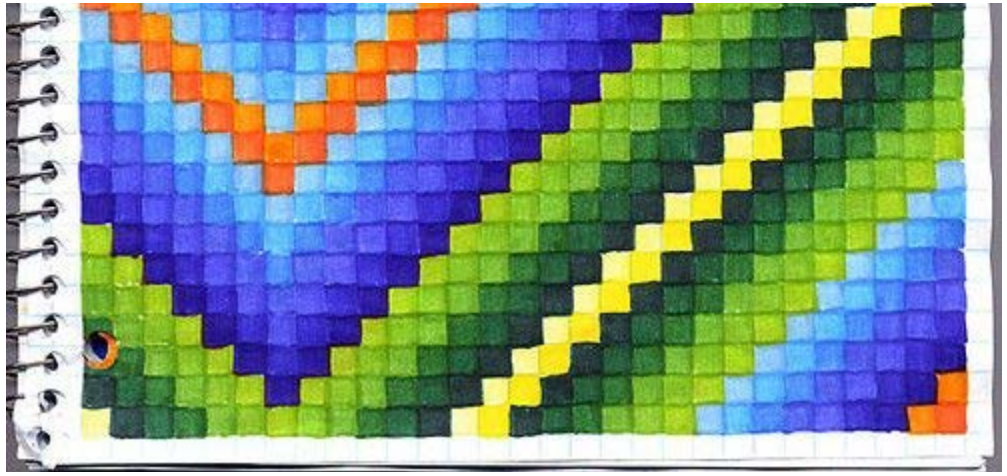

listeners

Listeners listen for events

- **Events** are often things users do, such a click a mouse, drag a mouse or press a key
- **mousePressed()** is one of these in p5. This will run the second the mouse is pressed
- **keyPressed()** is another. It will run the second a key is pressed



How do I draw stuff?



Functions

- **Function:** a named section of a program that does a specific task
  - Wraps up code in an easy to reference, repeatable way
  - Can be reused over and over
  - They are used by calling them

- **Parameter:** additional information you can give your function to change the starting state

Bake me a cake!

Baking a cake:
The action (function)

Chocolate: additional info that affects the actions (parameters)

Wait! I still don’t know how to draw stuff.

Shape

2D Primitives

arc()
ellipse()
line()
point()
quad()
rect()
triangle()

Attributes

ellipseMode()
noSmooth()
rectMode()
smooth()
strokeCap()
strokeJoin()
strokeWeight()

Curves

bezier()
bezierPoint()
bezierTangent()
curve()
curveTightness()
curvePoint()
curveTangent()

Vertex

beginContour()
beginShape()
bezierVertex()
curveVertex()
endContour()
endShape()
quadraticVertex()
vertex()

Color

Creating & Reading

alpha()
blue()
brightness()
color()
green()
hue()
lerpColor()
lightness()
red()
saturation()

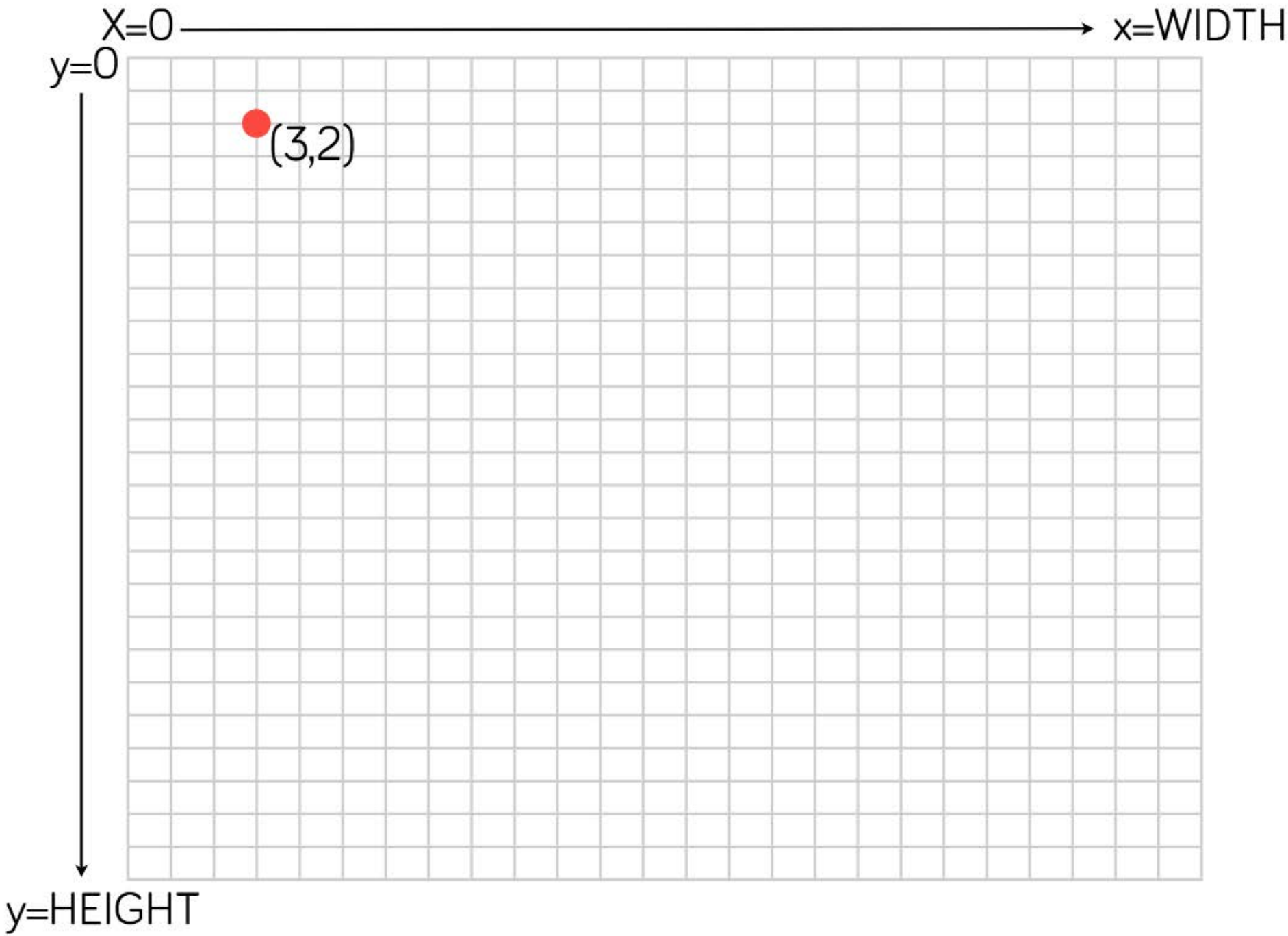
Setting

background()
clear()
colorMode()
fill()
noFill()
noStroke()
stroke()

Exercise: try drawing some stuff

How does positioning work?

How does positioning work?


X=0
x=WIDTH
y=0
(3,2)
y=HEIGHT

Coordinate Plane

- **X axis**: horizontal space that gets larger to the right
- **Y axis:** vertical space that gets larger as you go down

Exercise!

- Trying drawing a circle at x:600, y:800
- If you draw a circle at y:1000 is it higher or lower?

Can I incorporate interactivity?

Function structure

```
Bake_me_a_cake(chocolate, cherry){
        add_eggs();
}
```

- Name of function: Bake_me_a_cake
- Parenthesis: delineates it's a function, holds arguments
- Semicolon: end of line, move to the next thing

Mouse Positions

- **Variable**: a symbol used to stand in for a value. It can vary and change over time.
- **mouseX**: returns the position of the mouse x position
- **mouseY**: returns the position of the mouse y position

Exercise!

Draw a circle at mouseX and mouseY

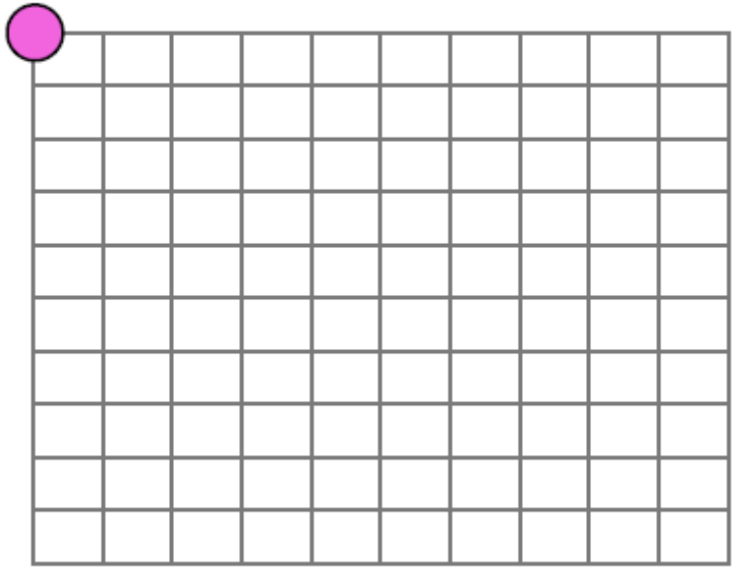
Variables

• Variables are useful for storing data that may change throughout the course of your app (e.g. your player's health)
• To create a variable, you have to tell the computer:
1. What kind of data you're storing (a number? a word?)
2. The name you're going to refer to it by

Some Variable Types

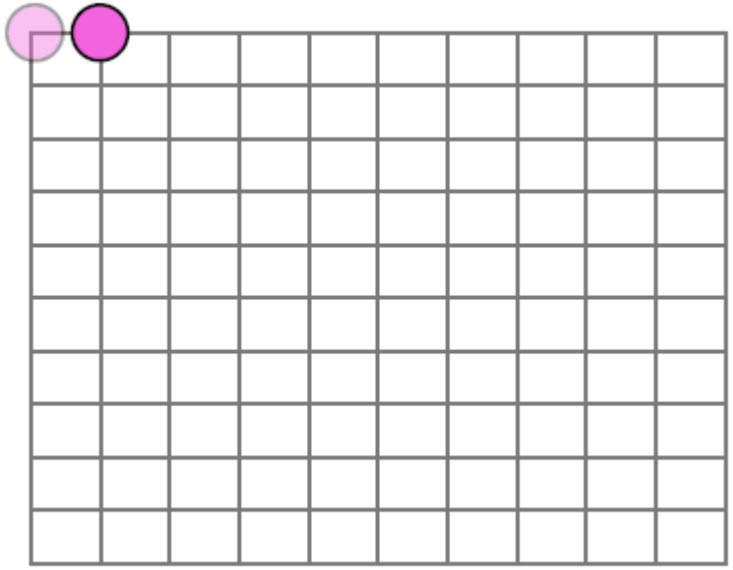
- **Float:** a decimal number (“I’m 5.4 feet tall.”)
- **Integer:** a whole number (“I’m 25 years old.”)
- **Boolean:** a true/false condition (“I’m not from California.”)
- **String:** text (“My name is Jane.”)
- **Char:** a single letter (A)

Movement



if our object starts at x=0, at time=0, and moves at a speed equal to 1 frame/sec, what will x equal at time=1?
How do you know?

Movement



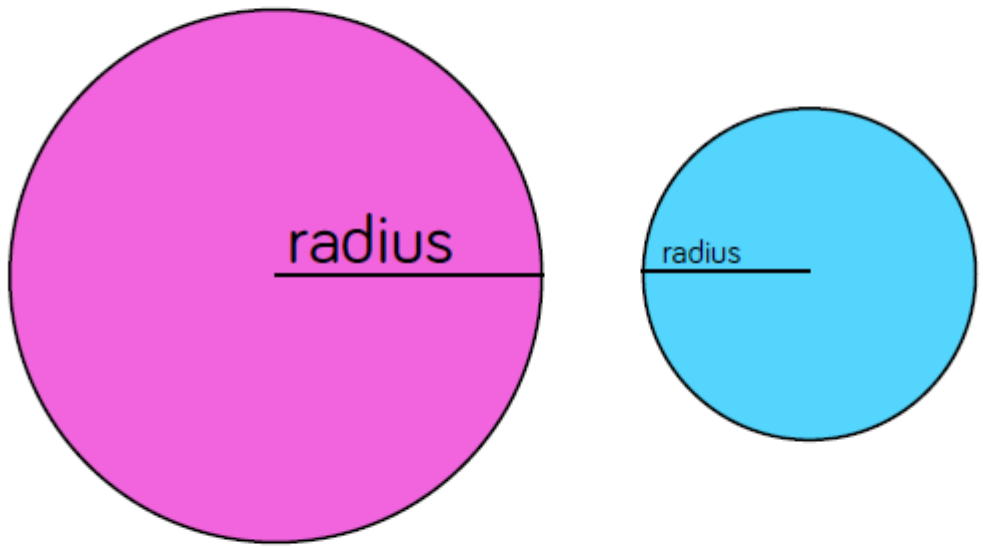
Movement
New position = old position + speed

Exercise!

Try drawing a circle that moves vertically down the screen.
Hint: you'll want a variable to hold the circle's position.
(Why?)

Collision: when one point is less than a certain distance from another point.

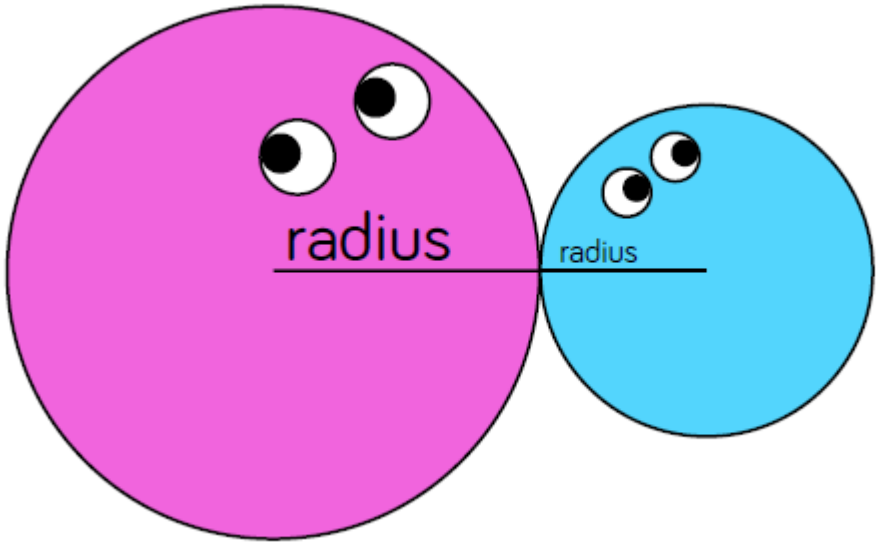
Have these circles collided yet?



Circle Collision

• If the distance between the center-points of the circles is less than or equal to the sum of their radii, they have collided!

• You can calculate distance in p5 with dist() function



Emergency kittens

How can I check if something is true?

If statements

- Consist of a condition and an action to take.
- Can have alternatives (if-else) and can put if statements inside of if-statements, too!

If I'm hungry, then I'll eat.

```
if (hungry) {
eat();
}
```

If statements

If I'm hungry, then I'll eat.

```
if (hungry) {
      eat();
}
```

If I'm hungry, then I'll eat.
Otherwise, I'll dance!

```
if (hungry) {
      eat();
} else {
      dance();
}
```

If I'm hungry, then I'll eat.
If I'm hungry and in the mood for pizza, I'll get pizza.
Otherwise, I'll dance!

```
if (hungry) {
      if (want_pizza) {
            eat(pizza);
      } else {
            eat(something_else);
      }
} else {
      dance();
}
```

Scope Operators!

{ }

- If you have one, you must have 2. They open and close.
- Variables declared inside a function scope are not visible to other scopes outside of the function.
- You can nest functions inside each other.
- There is more here but we will cover it soon...

Game States

State machine: *(simple explanation for now)*

- A state you can check to manage the state your game is in at any given point.
- A set of behaviors, UI, animations and events associated with a state
- Levels are simple states
- Examples are the start state of your game, the game playing state and the state your game is when it ends.

You can do this with a string in JavaScript simply.
var gameState = “start_game”;

Homework

Use the code here as a starting place:
(*https://github.com/phoenixperry/IGPED_Bootcamp/tree/master/ball_drop*) create a

Modify this game so the player's circle gets one point every time they successfully avoid a falling enemy's circle. The game ends when the enemy hits the player's circle. Slowly increase the circle speed and diameter until the player dies. Limit the max speed to a value within reason. The game must have a play button and a replay button.