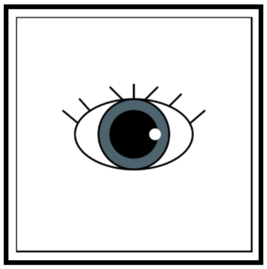
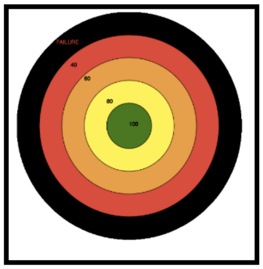
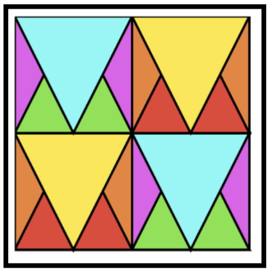
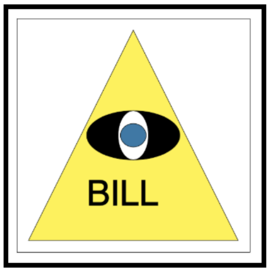
Logo, Emojis, and Character Design With p5.js









Here at the Code Next labs, we firmly believe that anyone can get started designing with code. So, here's a great introductory lesson to help you start! Students learn to compose shapes and colors into computational art, and create personalized graphics/logos and emojis!

Enjoy! (And let us know what you think of this curriculum here!)

Code Next

Logo, Emojis, and Character Design With p5.js

Project:
Logo, Emojis and Character Design with p5.js

Purpose /Context for this Day:
This is a fun design project where anyone can code in p5.js to compose shapes and colors into computational art! The ability to create and design personalized graphics/logos and emojis is not only fun, but also empowering in a digital world where we are surrounded by graphics and visual culture.

Goals:

- Explore basic coding concepts (coordinates, variables, and functions etc.)
- Plan a design on a paper grid, then transfer a design to the digital screen using code.
- Use a reference sheet of functions to plan a design.

Materials / Resources:

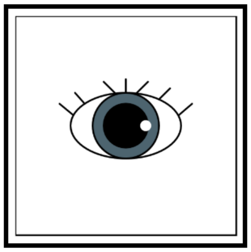
- Deck for Lesson
- Laptops with internet access
- Graph paper to plan your design (example)
- P5.js IDE (Free online p5.js software/code editor)
- CodeNext Starter Reference Guide
- P5.js Web Reference Guide
- Student examples

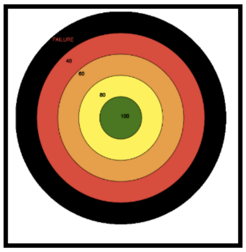
Essential Questions and Steps

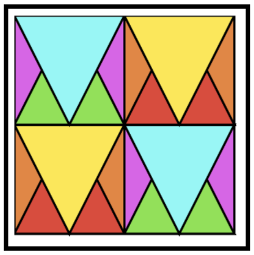
Essential Questions:

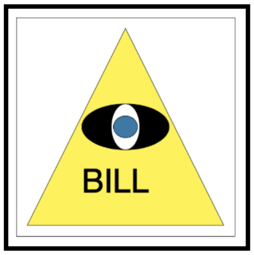
How do engineers and artists use code to create computational art?
How can I use code and graphics to create art that represents myself or my ideas?

Code Next Student Examples:



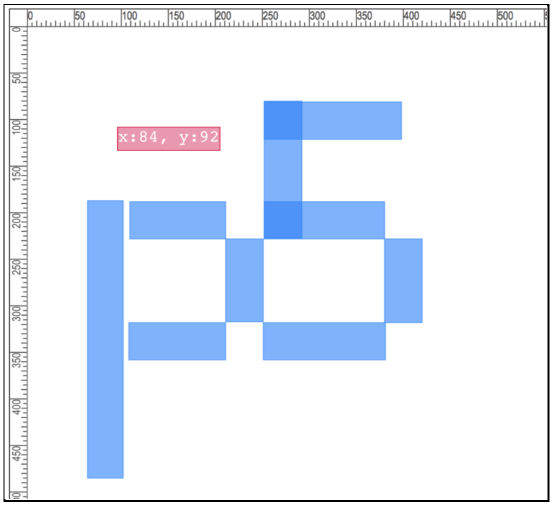




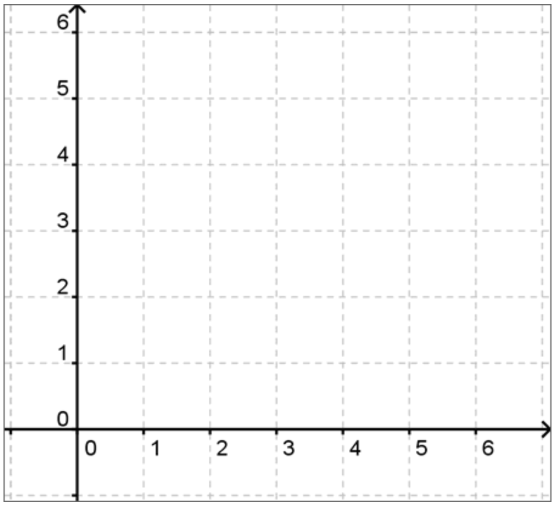


Step 1: Introduction to Canvas & Coordinates

The student work above was designed and coded on a ***canvas*** (basically a digital grid) of pixels with an x-axis and a y-axis. It's similar to a graph you would see in math class, except with two major differences. First, the lines aren't visible like graph paper, and second, the origin coordinate (0,0) is in the top left corner instead of the bottom left.



p5.js canvas with origin in top left



graph with origin at bottom left

The *canvas* has a x-axis that goes left to right, and a y-axis that goes top to bottom. On a regular graph you would see in math class, the origin is (0, 0) in the bottom left corner.

Coordinates are written (x, y). And x,y represents **variables**, or the **value** that tells the computer where you want the **function** (coded set of instructions) to start. In our digital canvas, the x increases from left to right. The y increases from top to bottom. The center point of a 400x400 pixel canvas would be at the coordinates (200, 200).

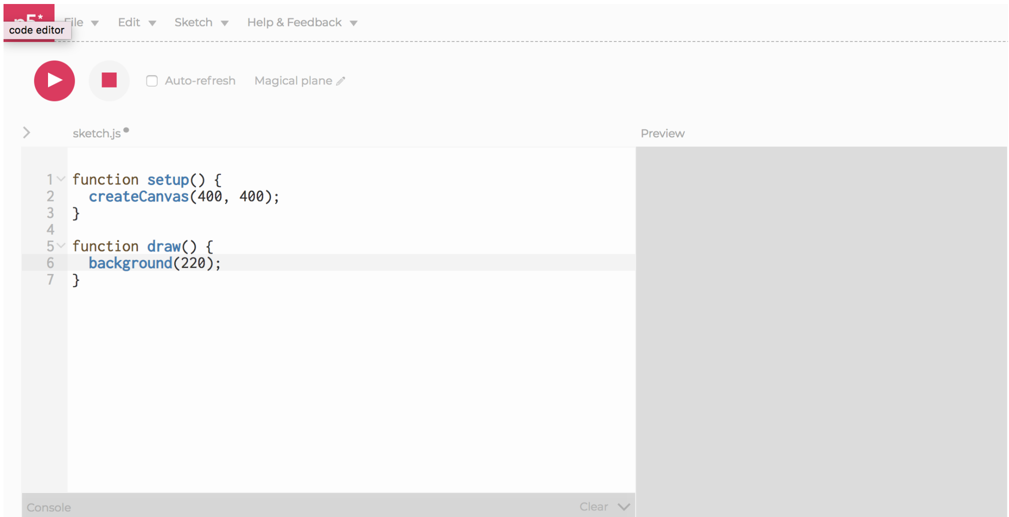
Step 2: Getting Started With p5.js

p5.js is an online coding site that gives you the ability to create computational art. After you learn and practice the basics, then you can plan out your own design on graph paper, and then plot out your lines of code.

Use this CodeNext starter p5 Reference Guide as you experiment. It is a helpful glossary of various functions for shapes, colors and text.

1. To get started, make sure you are connected to the internet and then click on this link. This will take you to the code editor with a blank 400x400pixel canvas.

Code Next



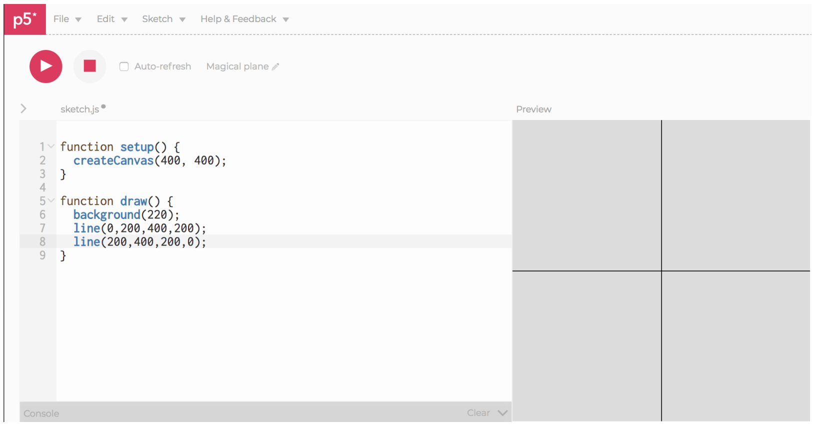
```
1  function setup() {
2    createCanvas(400, 400);
3  }
4
5  function draw() {
6    background(220);
7  }
```

2. The text on the lines 1-3 are functions that set up the blank 400x400 canvas. The code on lines 5-6 tell the computer to create a light grey background. If you change the value from 220 to 250, and click the pink "PLAY" button, your background will change to white.

{ 👆 👀When coding, grammar and punctuation are VERY important. If you have a spelling mistake, forget a semicolon or curly bracket, your code will not work and you will see an error message! ;}

Starting on line 7, but above the curly bracket **}** is where you will type in your lines of code. When you want to see the results of your code, click on the pink "PLAY" icon (located on the top left)**.** Your edits should appear on the canvas on the right side.

4. The first thing we will code is a **line**. To do this, we use the **line() function**. Again, a **function** is a line of code that contains a series of steps that the computer can understand. A line function looks like this: line(x1, y1, x2, y2); The computer understands this as instructions to create a line with the start and end points (x1,y1) and (x2,y2). For example, if you type in line(0,200,400,200); and click on the "PLAY" icon, you should see a line that goes horizontally across the entire canvas. If you type in another line of code: line(200,400,200,0); now, a line should also appear across half the canvas vertically.

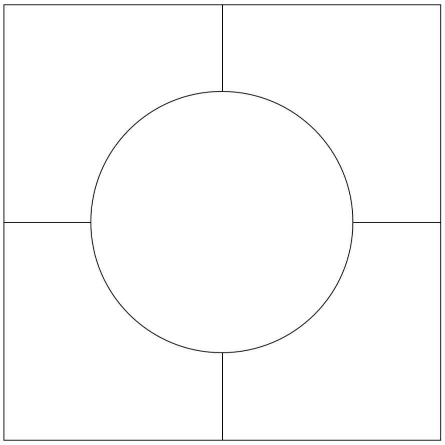


```
1  function setup() {
2    createCanvas(400, 400);
3  }
4
5  function draw() {
6    background(220);
7    line(0,200,400,200);
8    line(200,400,200,0);
9  }
```

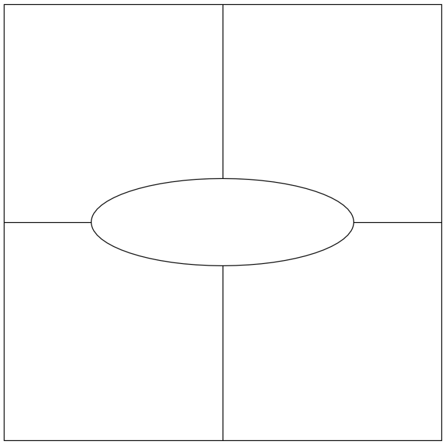
Google

5. Next, we will code a circular shape with the **ellipse function**. An **ellipse** is a rounded shape, (sometimes a circle, sometimes oval). The ellipse function looks like this: ellipse(x, y, width, height);
The computer understands this as steps to draw a rounded shape with center point at: (x,y) and with (width and height).

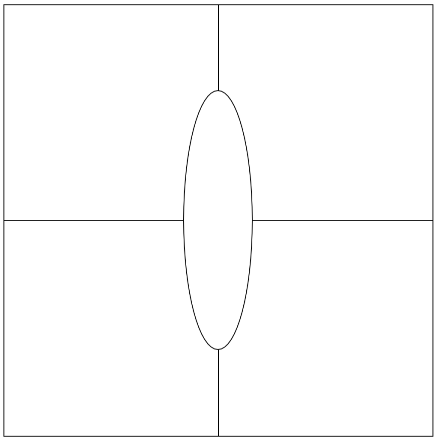
If you type in: ellipse(200, 200, 300, 300); you should get a circle that is in the middle of your canvas that is 300x300 pixels. Try it, and then change some of the values. How would you move your circle UP or DOWN? How about to make it wider, like a lemon, or long and skinny like a bunny ear?



ellipse(200, 200, 300, 300);

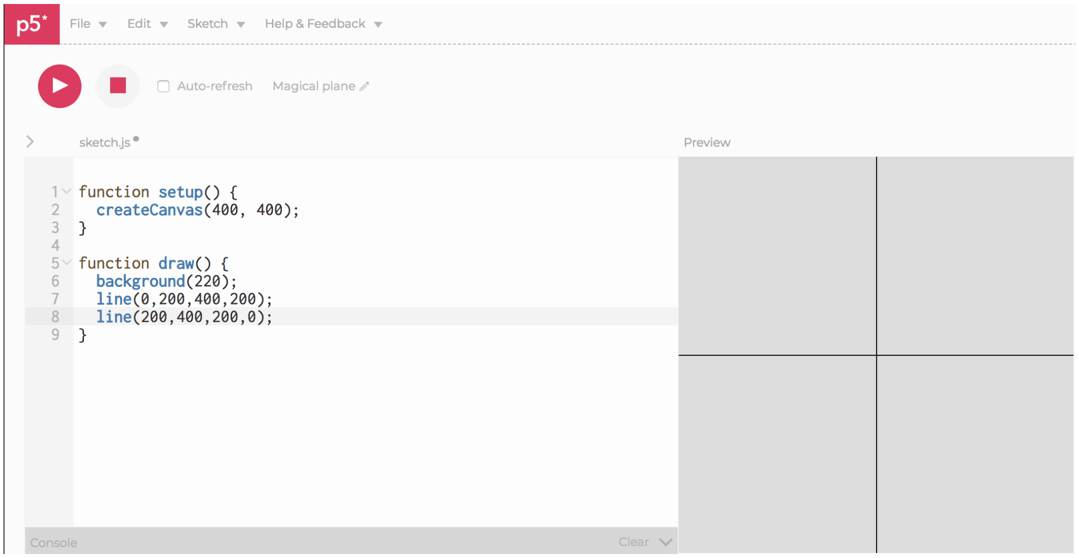


ellipse(200, 200, 300, 100);



ellipse(200, 200, 80, 300);

6. Using the p5.js Reference Guide , try a rectangle function. rect(x, y, width, height);
Note that the origin for the ellipse is placed by its center point, but rectangle is placed by its top left corner. So the function: rect(150,150,100,100); should create a 100x100pixel square in the middle of your canvas. Experiment again with changing the dimensions and position of a rectangle.

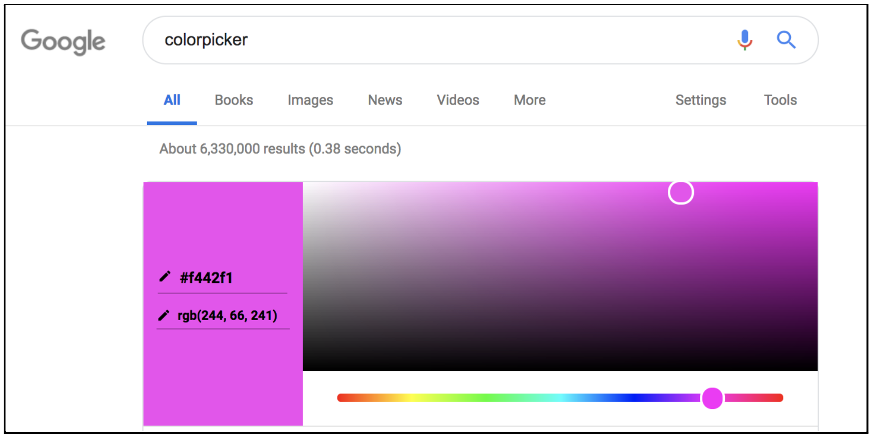


```
function setup() {
  createCanvas(400, 400);
}

function draw() {
  background(220);
  line(0,200,400,200);
  line(200,400,200,0);
}
```

Code Next

7. Time to add color!
To add color to the inside of a shape use the function: fill('#000000'); The #000000 represents a color code. You can find any color code you want by doing a Google Web search for, "color picker" and moving around the slide-bar and pointer. Once you find the color you like, you can copy and paste the color code. In the magenta below it would be '#f442f1'. **Note:** The fill function must come before the shape function. Use the p5 Reference Guide and try to add color to your **outlines**, and then try to make the width of your lines thicker or thinner (**stroke weight**).

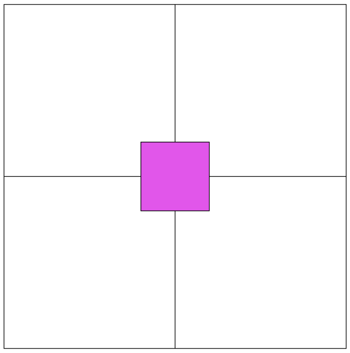


colorpicker

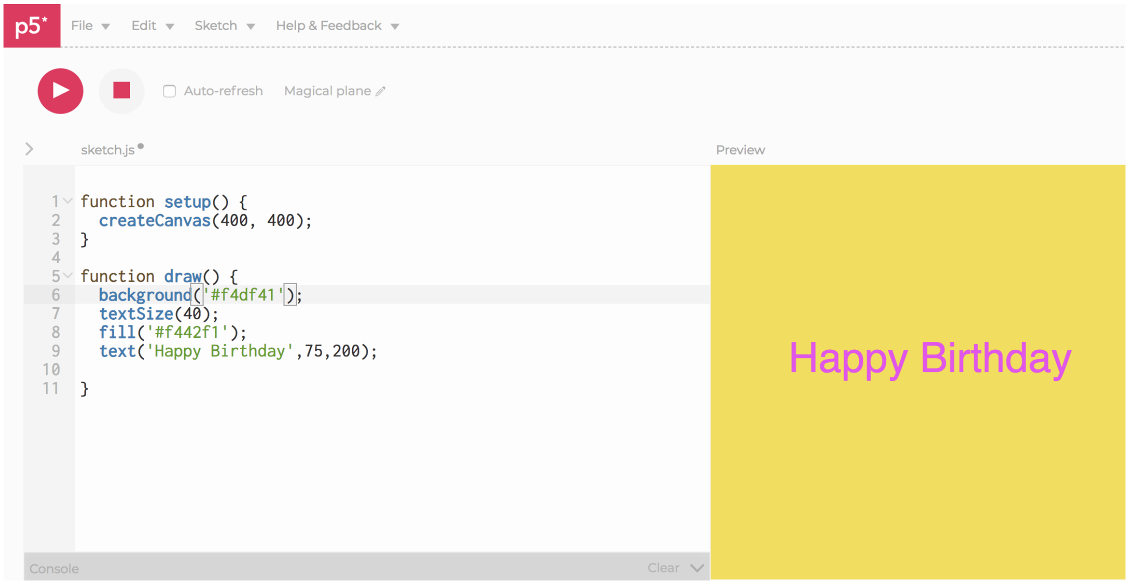
About 6,330,000 results (0.38 seconds)

#f442f1

rgb(244, 66, 241)



8. Next you can add text using the function text('Text', x, y); and you can change the text size with textSize(number); The computer understands this as instructions to write the text in quotes starting at coordinate (x,y). If you type in the code below, you should get pink text on a yellow background that spells out 'Happy Birthday' in the middle of your canvas.



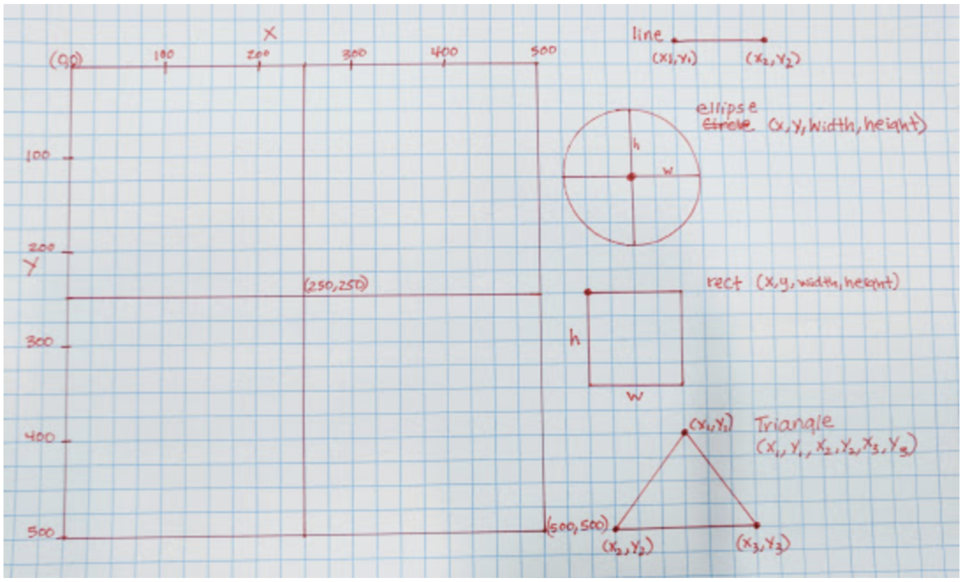
```
function setup() {
  createCanvas(400, 400);
}

function draw() {
  background('#f4df41');
  textSize(40);
  fill('#f442f1');
  text('Happy Birthday',75,200);

}
```

Happy Birthday

9. When you are ready to plan out your own design, it can be helpful to use graph paper and pencil to plan out your designs. You can create a canvas template using graph like the example below with each square representing 25 pixels.



Step 3: Get Creative and Experiment!

Now that you have had an introduction to basic **functions** and writing lines of code to create shapes, color and text, it's your turn to experiment! Try something! Make mistakes! If something doesn't work, or you get an error message, you can always **debug your code** by checking the reference guide, asking a friend to take a closer look or doing a Google search to find the answer. (Even professional engineers have to troubleshoot and debug every day). **Have fun, and Happy Coding!** 👏😎👾