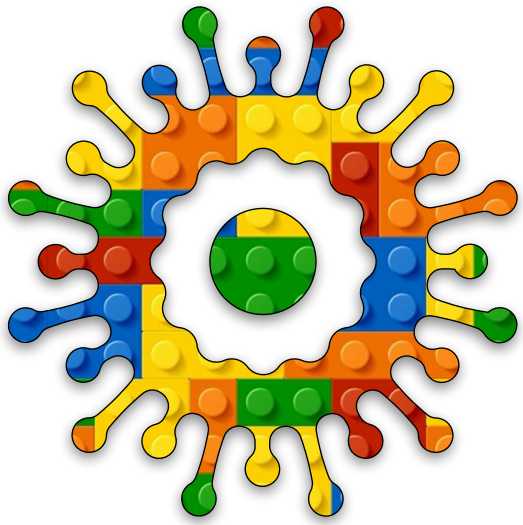
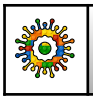
Creative Coding Module A Unit #3 adding RGB





Content

Module A Unit #3 adding RGB

Sketch A3.1	RGB
Sketch A3.2	alpha
Sketch A3.3	random colour alpha value
Sketch A3.4	no stroke
Sketch A3.5	the weight of the stroke
Sketch A3.6	colouring the lines

Introduction to adding RGB

This unit looks at RGB colour, by that we mean red, green, and blue. This is how you can create a wide range of colours by mixing these primary colours; in fact, you can create over 16 million colours.

Think in terms of light rather than paints. If you have all the red, all the green, and all the blue, you get white; if you have none of them, you get black, and every combination in between. The amount of a single colour is between 0 - 256.

Key concepts covered in this unit:

- RGB colour
- transparency
- noStroke()
- stroke()
- strokeWeight()

RGB colour

We can represent the colour not just in words— red, green, or blue — but also using their RGB values. Each colour has some red, some green, and some blue. One reason for using this system of creating the colours is that we can manipulate the values mathematically. For instance, the following colours are given as RGB values. See Fig. 1 below.

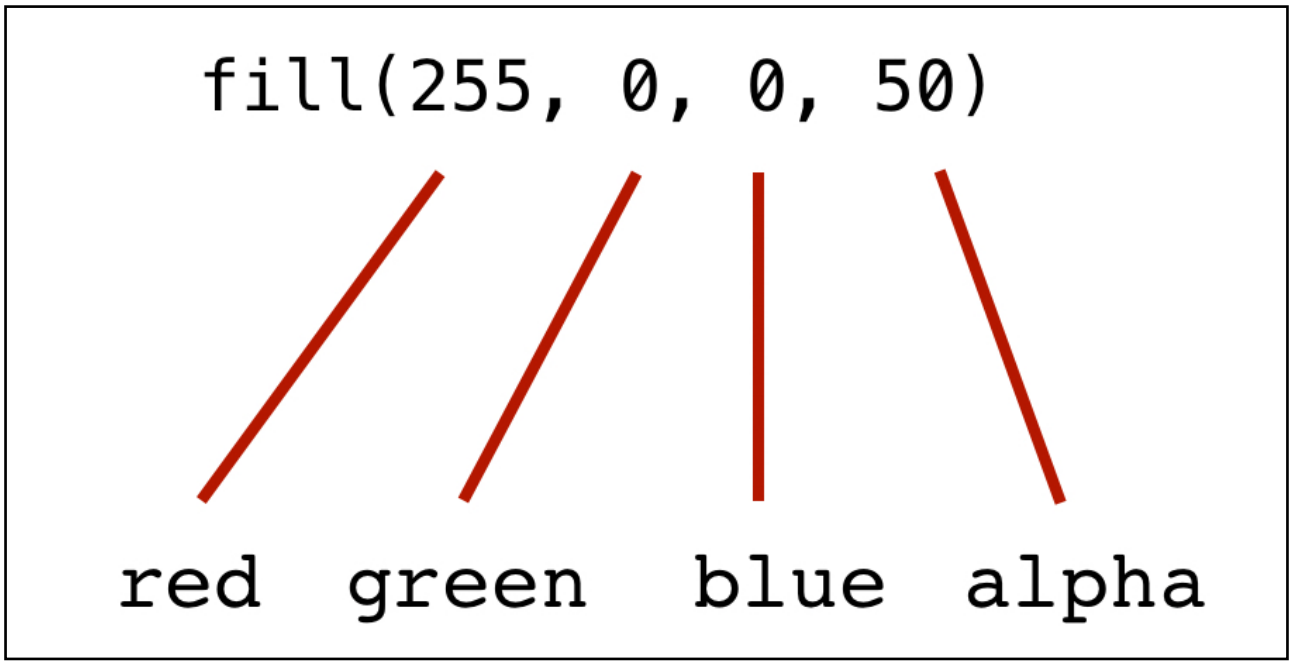
`fill(255, 0, 0)`	This gives us red with no green and no blue
`fill(0, 255, 0)`	This gives us green with no red or blue
`fill(0, 0, 255)`	This gives us blue with no red or green
`fill(255, 255, 255)`	This gives us white, same as `fill(255)`
`fill(0, 0, 0)`	This gives us black, same as `fill(0)`

RGB transparency

You can create any colour by mixing the numbers as you do with mixing paint or light. The three numbers are called `arguments`; you can add a fourth argument, which is the `alpha` or transparency value. The range is between `0` and `255`. See `Fig. 1` below.

`fill(255, 0, 0, 255)`	This gives us no transparency at all, even though it should be red, it will appear opaque
`fill(255, 0, 0, 0)`	This gives us no transparency at all, even though it should be red, it will appear completely transparent
`fill(255, 0, 0, 50)`	Gives us some red but you can still see through it

Figure 1: fill() function



Sketch A3.1 RGB

❗ Start a new sketch

Drawing three overlapping circles, each with a separate colour. In the previous unit, we used names for the colours used. The background was called `'lightgrey'`. That is one way of using colours in your sketch; alternatively, we can use numerical values. Here we are using the RGB values for the background and the colours of the circles. I have highlighted the code in blue. Notice that I have given the `background()` a value of `220`, which is a single colour value, meaning it is either black (value of `0`) or white (value of `255`) and everything in between as shades of grey.

```
function setup()
{
  createCanvas(400, 400)
  background(220)
}

function draw()
{
  fill(255, 0, 0)
  circle(175, 150, 100)
  fill(0, 255, 0)
  circle(225, 150, 100)
  fill(0, 0, 255)
  circle(200, 200, 100)
}
```

🗒️ Notes

Each circle will have its own colour fill.

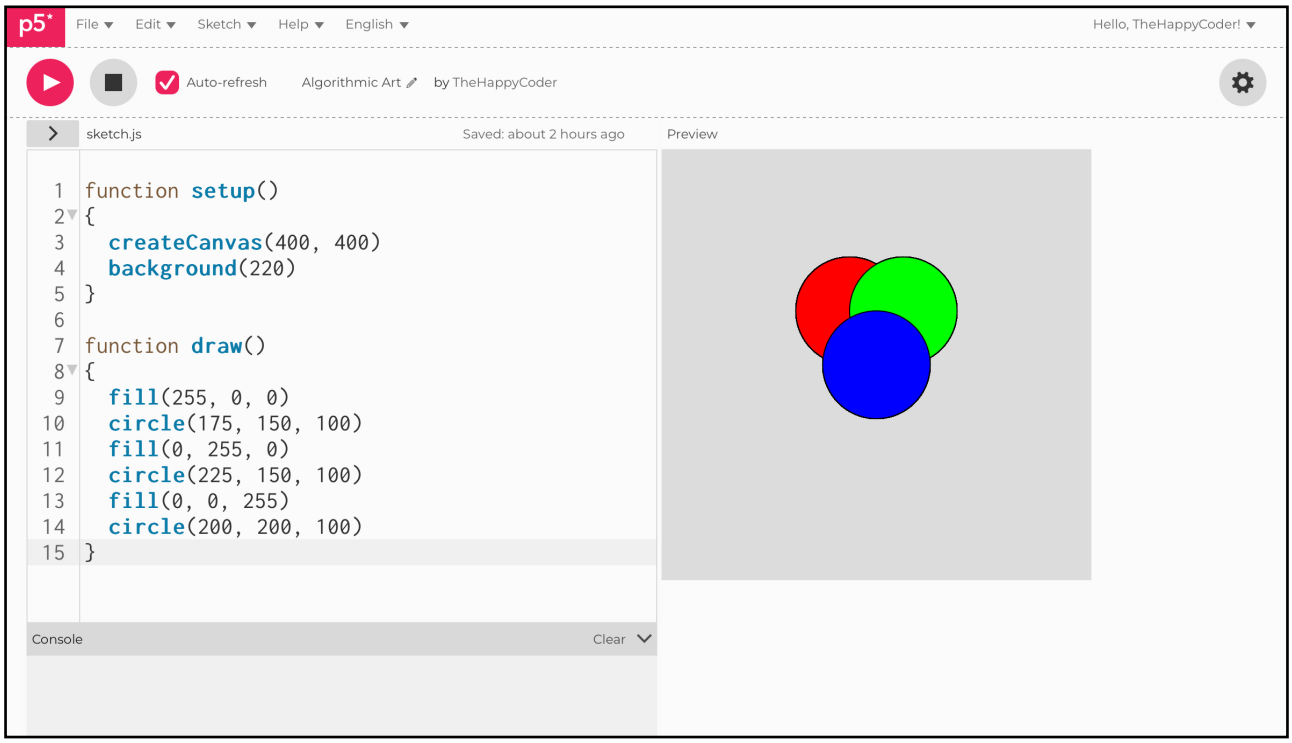
🌻 Challenge

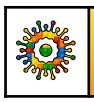
Remove one of the `fill()` functions and see what happens.

🛠️ Code Explanation

`background(220)`	The background has a grey colour
`fill(255, 0, 0)`	This gives you a red coloured circle, where the red value = 255, green value = 0, and blue has a value = 0
`fill(0, 255, 0)`	This gives you a green coloured circle, where the red value = 0, green value = 255, and blue has a value = 0
`fill(0, 0, 255)`	This gives you a blue coloured circle, where the red value = 0, green value = 0, and blue has a value = 255

Figure A3.1


p5*
File
Edit
Sketch
Help
English
Hello, TheHappyCoder!
Auto-refresh
Algorithmic Art
by TheHappyCoder
sketch.js
Saved: about 2 hours ago
Preview
1 function setup()
2 {
3 createCanvas(400, 400)
4 background(220)
5 }
6
7 function draw()
8 {
9 fill(255, 0, 0)
10 circle(175, 150, 100)
11 fill(0, 255, 0)
12 circle(225, 150, 100)
13 fill(0, 0, 255)
14 circle(200, 200, 100)
15 }
Console
Clear



Sketch A3.2 alpha

Adding a bit of alpha by adding a fourth argument, we give the circles an `alpha` value of `50`.

```
function setup()
{
  createCanvas(400, 400)
  background(220)
}

function draw()
{
  fill(255, 0, 0, 50)
  circle(175, 150, 100)
  fill(0, 255, 0, 50)
  circle(225, 150, 100)
  fill(0, 0, 255, 50)
  circle(200, 200, 100)
}
```

🗒️ Notes

The overlapping circles are now more translucent, with the help of some alpha.

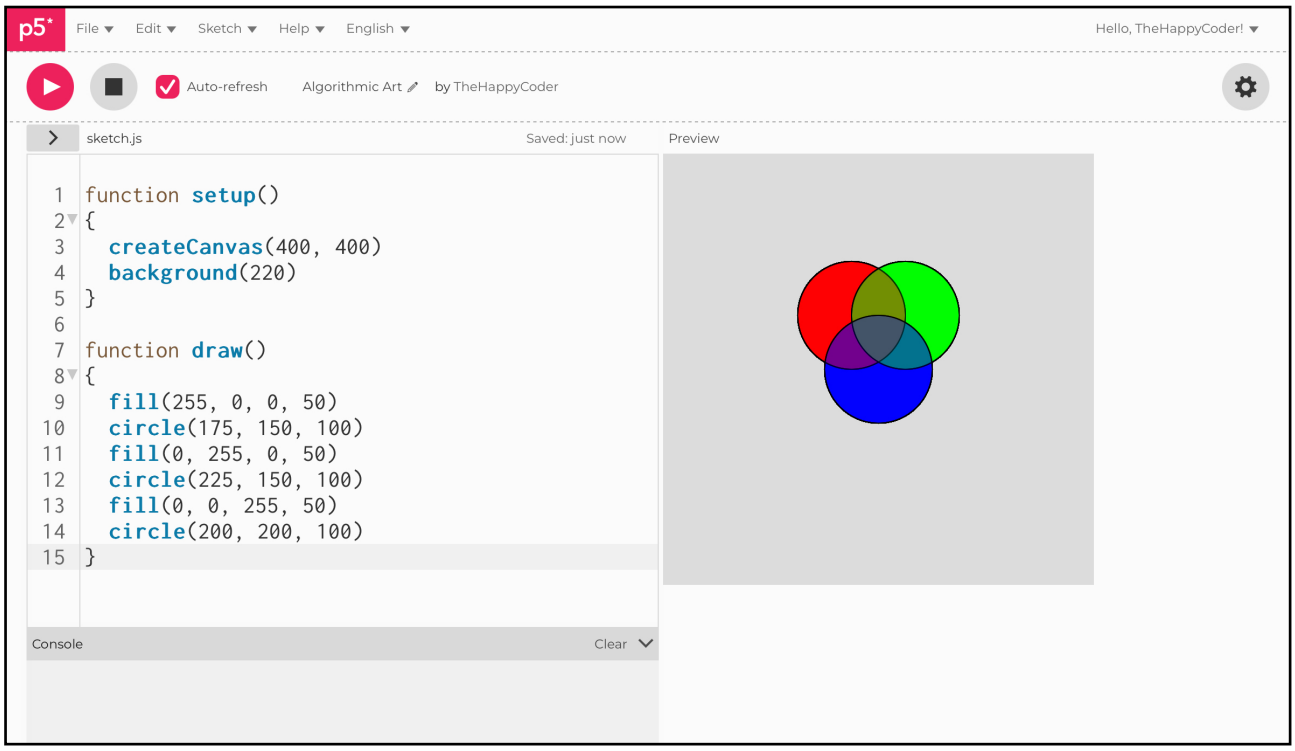
🌻 Challenge

Change the values of alpha.

🛠️ Code Explanation

`fill(255, 0, 0, 50)`	Red with plenty of transparency
`fill(0, 255, 0, 50)`	Green with plenty of transparency
`fill(0, 0, 255, 50)`	Blue with plenty of transparency

Figure A3.2


p5*
File
Edit
Sketch
Help
English
Hello, TheHappyCoder!
Auto-refresh
Algorithmic Art
by TheHappyCoder
sketch.js
Saved: just now
Preview
1 function setup()
2 {
3 createCanvas(400, 400)
4 background(220)
5 }
6
7 function draw()
8 {
9 fill(255, 0, 0, 50)
10 circle(175, 150, 100)
11 fill(0, 255, 0, 50)
12 circle(225, 150, 100)
13 fill(0, 0, 255, 50)
14 circle(200, 200, 100)
15 }
Console
Clear

Sketch A3.3 random colour alpha value

We are now going to go mad with the variables, making variables of nearly everything. The alpha is the fourth argument for colour and it is the amount of transparency from 0 (transparent) to 255 (opaque).

```
let x = 0
let y = 0
let r = 0
let g = 0
let b = 0
let a = 0
let d = 0

function setup()
{
  createCanvas(400, 400)
  background(220)
}

function draw()
{
  x = random(400)
  y = random(400)
  r = random(255)
  g = random(255)
  b = random(255)
  a = random(255)
  d = random(100)
  fill(r, g, b, a)
  circle(x, y, d)
}
```

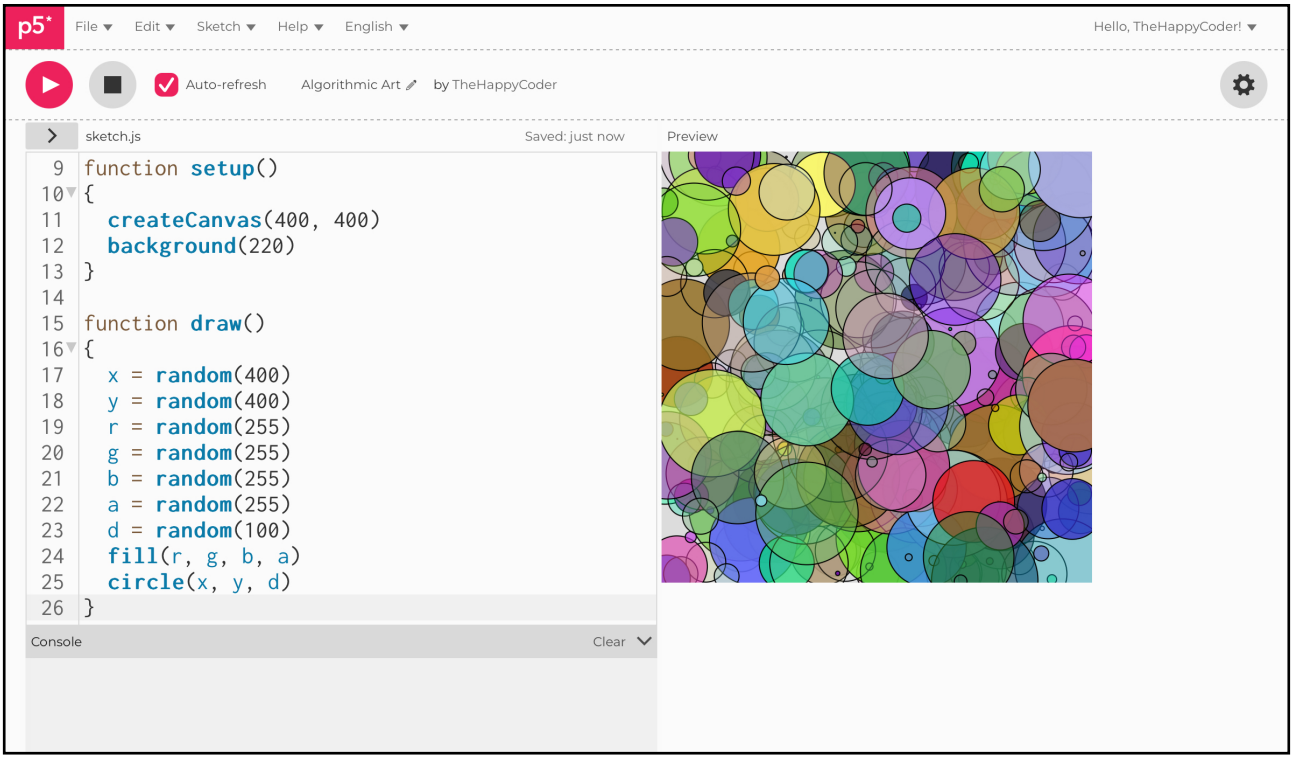
🗒️ Notes

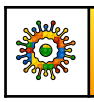
You wouldn't generally use single letters to name variables, but in this case, it is simple enough and intuitive.

🛠️ Code Explanation

`x = random(400)`	Random x position of circle
`y = random(400)`	Random y position of circle
`r = random(255)`	Random red value
`g = random(255)`	Random green value
`b = random(255)`	Random blue value
`a = random(255)`	Random alpha
`d = random(100)`	Random diameter
`fill(r, g, b, a)`	Fill each circle with one set of random values
`circle(x, y, d)`	Draw a circle on each loop of the draw() function, at random positions and diameters

Figure A3.3


p5*
File
Edit
Sketch
Help
English
Hello, TheHappyCoder!
Auto-refresh
Algorithmic Art
by TheHappyCoder
sketch.js
Saved: just now
Preview
9 function setup()
10 {
11 createCanvas(400, 400)
12 background(220)
13 }
14
15 function draw()
16 {
17 x = random(400)
18 y = random(400)
19 r = random(255)
20 g = random(255)
21 b = random(255)
22 a = random(255)
23 d = random(100)
24 fill(r, g, b, a)
25 circle(x, y, d)
26 }
Console
Clear



Sketch A3.4 no stroke

We can remove the line around the circle with the function `noStroke()`.

```
let x = 0
let y = 0
let r = 0
let g = 0
let b = 0
let a = 0
let d = 0

function setup()
{
  createCanvas(400, 400)
  background(220)
}

function draw()
{
  noStroke()
  x = random(400)
  y = random(400)
  r = random(255)
  g = random(255)
  b = random(255)
  a = random(255)
  d = random(100)
  fill(r, g, b, a)
  circle(x, y, d)
}
```

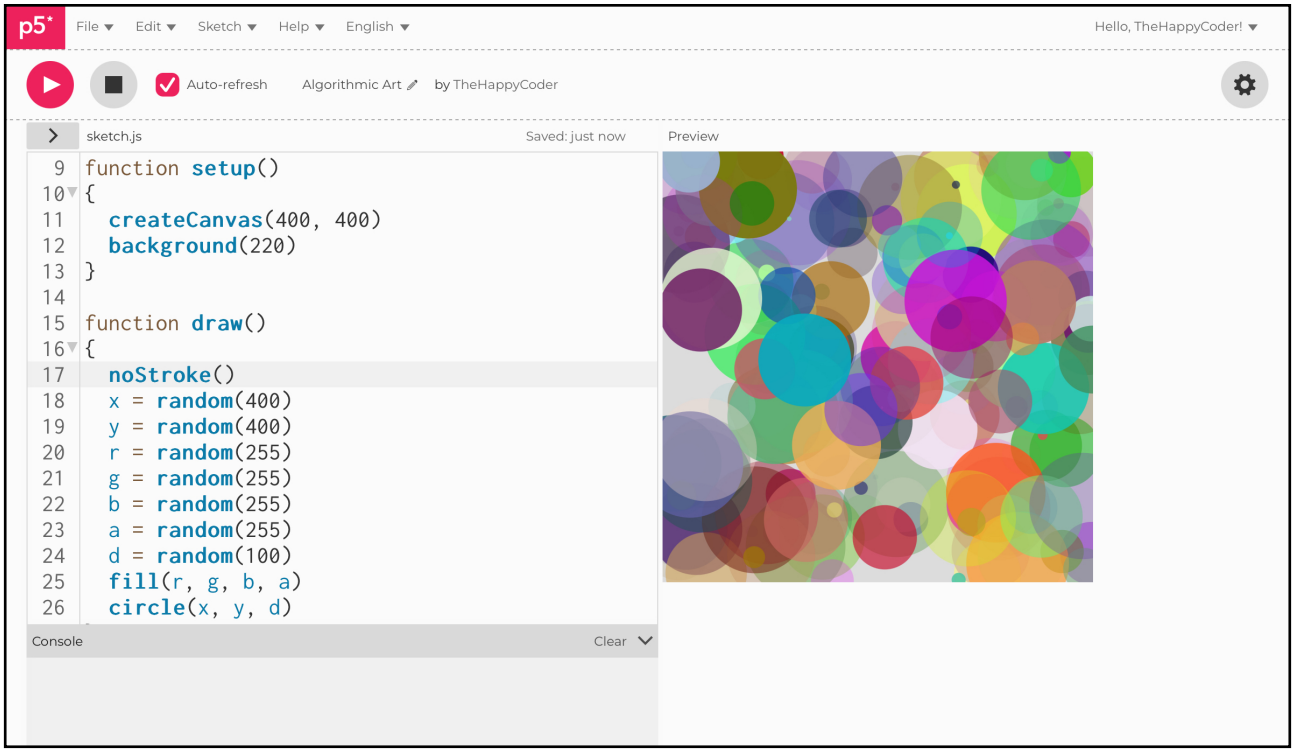
🗒️ Notes

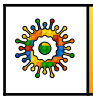
Having `noStroke()` and `noFill()` will effectively remove the circle from the canvas.

🛠️ Code Explanation

`noStroke()`	Removes any lines around shapes

Figure A3.4


p5*
File
Edit
Sketch
Help
English
Hello, TheHappyCoder!
Auto-refresh
Algorithmic Art
by TheHappyCoder
sketch.js
Saved: just now
Preview
9 function setup()
10 {
11 createCanvas(400, 400)
12 background(220)
13 }
14
15 function draw()
16 {
17 noStroke()
18 x = random(400)
19 y = random(400)
20 r = random(255)
21 g = random(255)
22 b = random(255)
23 a = random(255)
24 d = random(100)
25 fill(r, g, b, a)
26 circle(x, y, d)
Console
Clear



Sketch A3.5 the weight of the stroke

❗ Remove `noStroke()` and replace with `strokeWeight()`

We can alter the thickness of the lines with the function `strokeWeight()`, the default is a thickness of 1 pixel, so you could use any value you like, but I recommend only using a value up to 5.

```
let x = 0
let y = 0
let r = 0
let g = 0
let b = 0
let a = 0
let d = 0

function setup()
{
  createCanvas(400, 400)
  background(220)
}

function draw()
{
  strokeWeight(random(5))
  x = random(400)
  y = random(400)
  r = random(255)
  g = random(255)
  b = random(255)
  a = random(255)
  d = random(100)
  fill(r, g, b, a)
  circle(x, y, d)
```

```
}
```

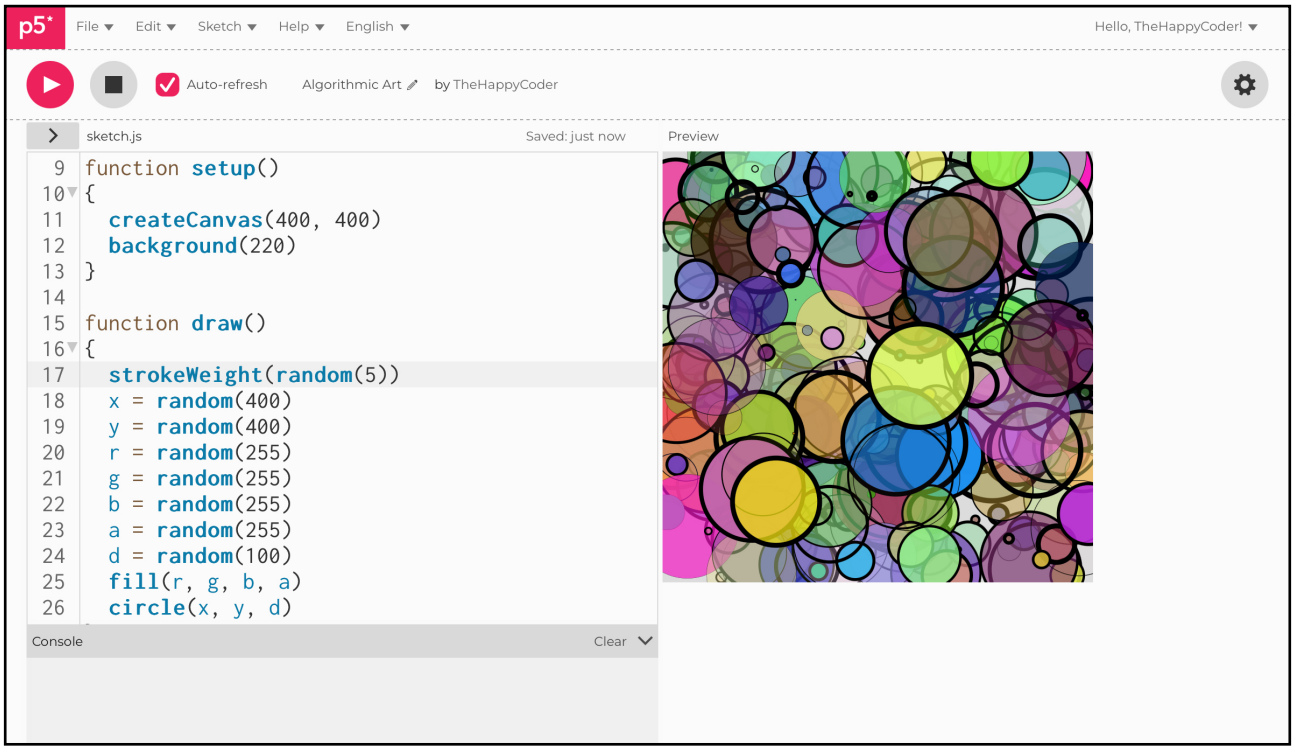
🌻 Challenge

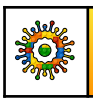
Try different ranges of random `strokeWeight()`.

🛠️ Code Explanation

`strokeWeight(random(5))`	We can put the random function inside the brackets rather than create another variable

Figure A3.5


p5*
File
Edit
Sketch
Help
English
Hello, TheHappyCoder!
Auto-refresh
Algorithmic Art
by TheHappyCoder
sketch.js
Saved: just now
Preview
9 function setup()
10 {
11 createCanvas(400, 400)
12 background(220)
13 }
14
15 function draw()
16 {
17 strokeWeight(random(5))
18 x = random(400)
19 y = random(400)
20 r = random(255)
21 g = random(255)
22 b = random(255)
23 a = random(255)
24 d = random(100)
25 fill(r, g, b, a)
26 circle(x, y, d)
Console
Clear



Sketch A3.6 colouring the lines

❗ Replace `strokeWeight()` with a function called `stroke()`

We can also add colour (even add alpha) to the lines using `stroke()`, it works exactly the same as the `fill()` function.

```
let x = 0
let y = 0
let r = 0
let g = 0
let b = 0
let a = 0
let d = 0

function setup()
{
  createCanvas(400, 400)
  background(220)
}

function draw()
{
  stroke(r, g, b)
  x = random(400)
  y = random(400)
  r = random(255)
  g = random(255)
  b = random(255)
  a = random(255)
  d = random(100)
  fill(r, g, b, a)
  circle(x, y, d)
```

```
}
```

🗒️ Notes

The colour of the lines is the colour of the last `fill(r, g, b)` circle drawn.

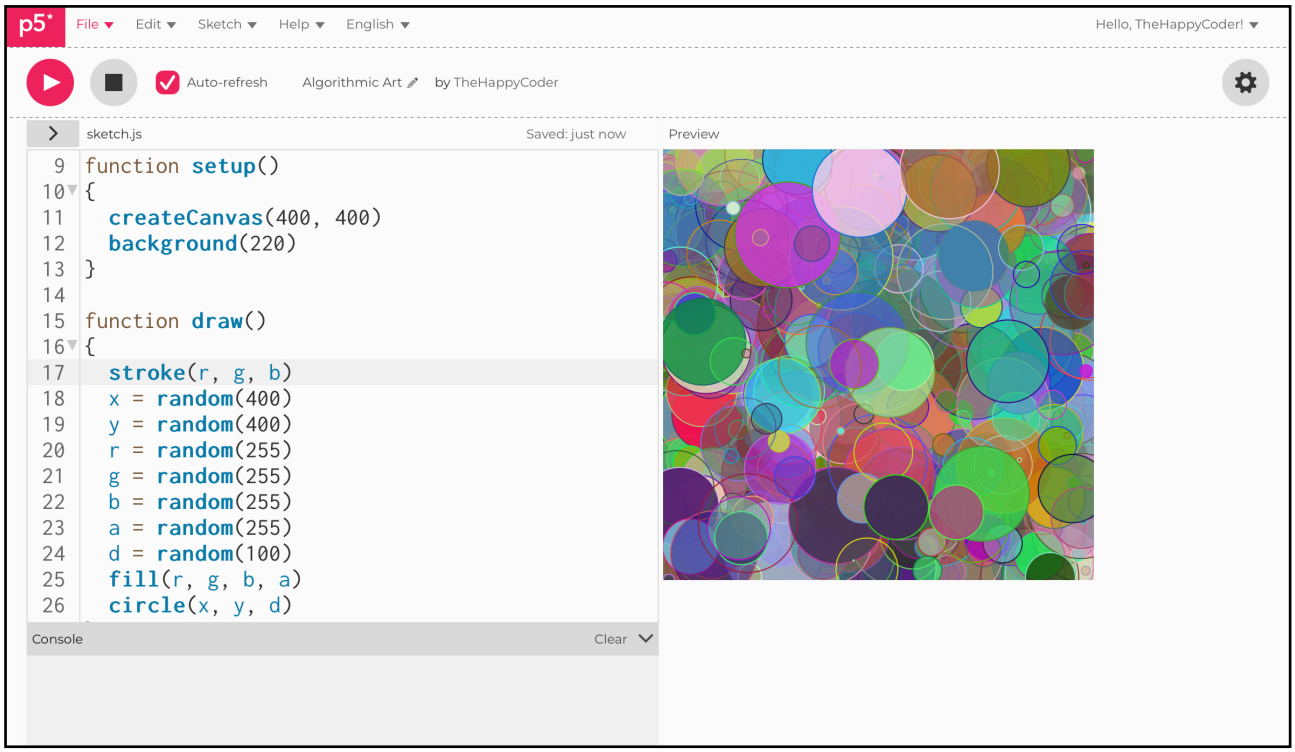
🌻 Challenges

1. Add random `strokeWeight()` as well.
2. Move `stroke(r, g, b)` to the line just before `fill(r, g, b, a)`.

🛠️ Code Explanation

`stroke(r, g, b)`	As with the fill() function we can give it up to four arguments, red, green, blue and alpha

Figure A3.6


p5*
File
Edit
Sketch
Help
English
Hello, TheHappyCoder!
Auto-refresh
Algorithmic Art
by TheHappyCoder
sketch.js
Saved: just now
Preview
9 function setup()
10 {
11 createCanvas(400, 400)
12 background(220)
13 }
14
15 function draw()
16 {
17 stroke(r, g, b)
18 x = random(400)
19 y = random(400)
20 r = random(255)
21 g = random(255)
22 b = random(255)
23 a = random(255)
24 d = random(100)
25 fill(r, g, b, a)
26 circle(x, y, d)
Console
Clear

Next. . .

Unit #4 looks at drawing what should be a simple line. There's a lot you can do with a line, and it requires a bit more thought than drawing a circle.