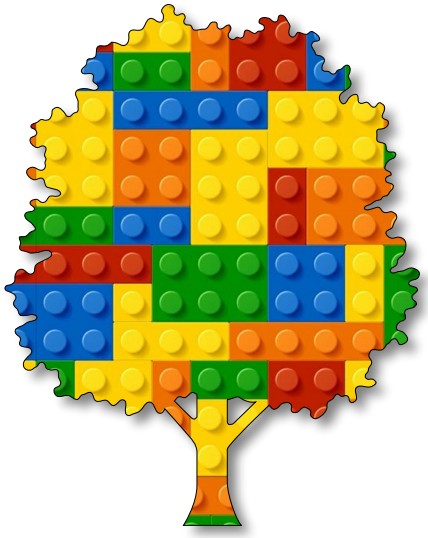
Algorithmic Art

Module B

Unit #5

HSB colours



Content

Module B Unit #5 HSB colours

Introducing HSB colours

HSB is another colour format, where HSB stands for Hue, Saturation, and Brightness. We can only access this format using the colourMode() function. It creates pleasant colours but is less intuitive than RGB. The Hue has values from 0 to 360, Saturation has values from 0 to 100, and Brightness also has values from 0 to 100.

Sketch B5.1 using HSB for the colour

HSB (or HSL) stands for Hue, Saturation, and Brightness (or Lightness).

中 Hue

The first argument is the hue, which is on a colour circle from `0` to `360`, where `0` is red, `120` is green, and `240` is blue.

中 Saturation

The second argument is the saturation, which is a percentage from `0` to `100`. It is the amount of colour, so `0` is white to `100` being full colour.

中 Brightness

The third argument is the level of brightness, which is from `0` to `100` also, where `0` is completely dark, and `100` fully bright.

❗ Start a new sketch and add in the highlighted lines of code. For this, we have to change the `colorMode()` to HSB. Here we get a green background and a blue circle.

```
function setup()
{
  createCanvas(400, 400)
  colorMode(HSB)
}

function draw()
{
  background(0, 50, 100)
  fill(200, 50, 100)
  circle(100, 100, 100)
}
```

Notes

The default is RGB, so that is why we call the `colorMode()` for HSB. You can switch back by calling the `colorMode(RGB)`. HSB produces some very pleasing colours.

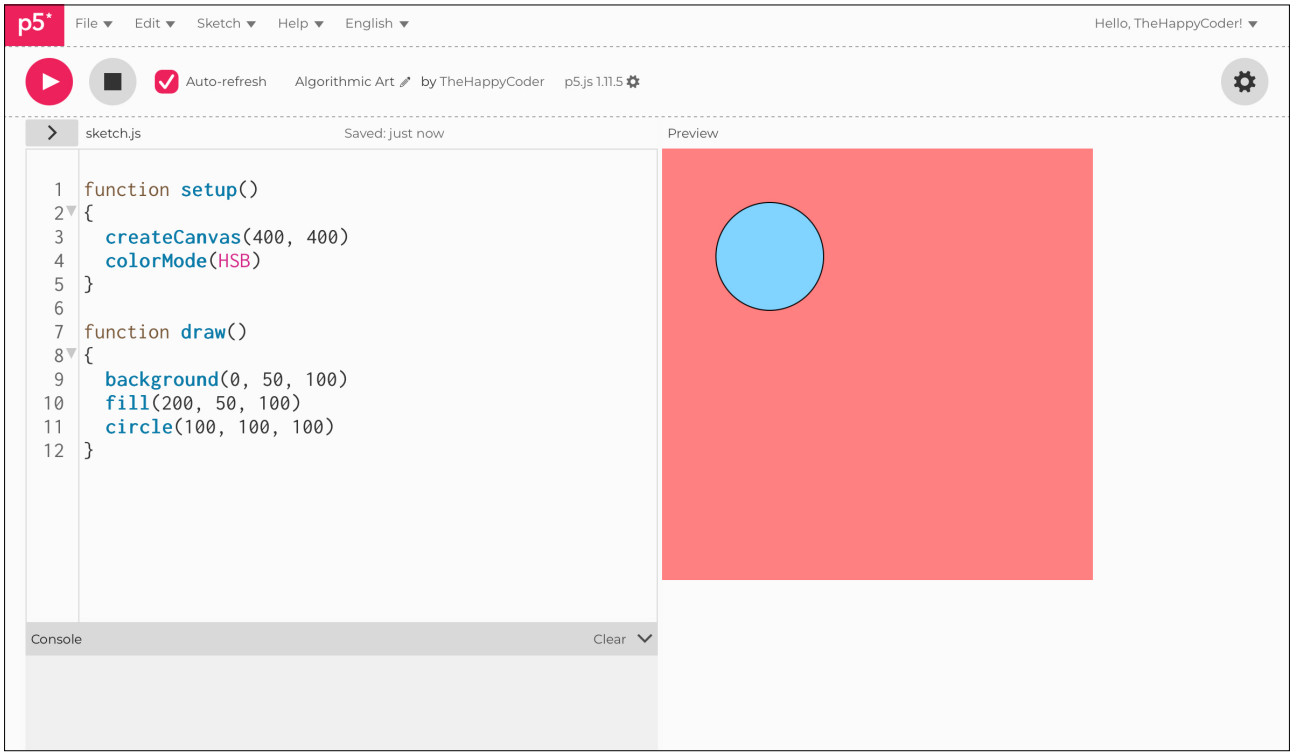
Challenge

Play with the numbers.

Code Explanation

`colorMode(HSB)`	Sets the mode for HSB values
`background(0, 50, 100)`	A pale red background
`fill(200, 50, 100)`	A pale blue circle

Figure B5.1



Sketch B5.2 HSB colour chart

❗ a newish sketch.
We iterate through the colours 10 steps at a time from 0 to 360. This is the first argument. Notice that we have a canvas of 360x400 for a change.

```
function setup()
{
  createCanvas(360, 400)
  colorMode(HSB)
  noStroke()
}

function draw()
{
  for (let i = 0; i < 360; i++)
  {
    fill(i, 100, 100)
    rect(i, 1, 1, 400)
  }
}
```

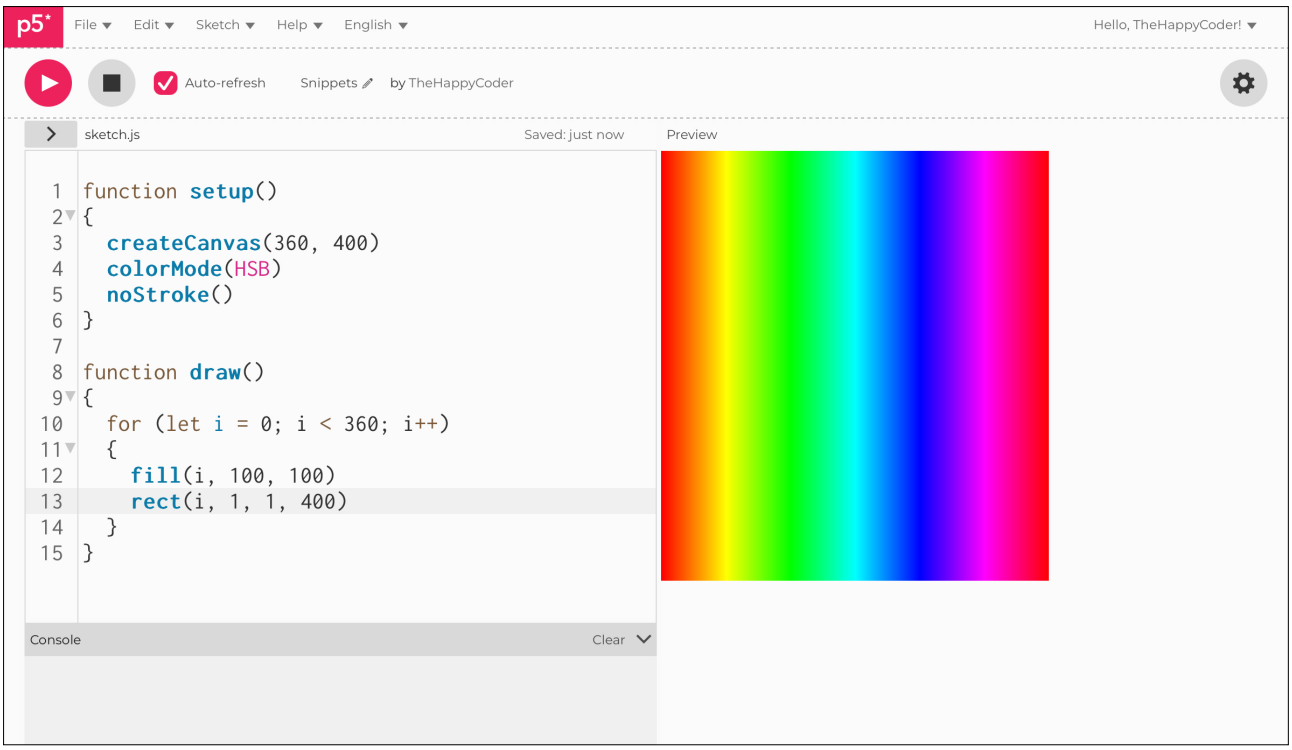
Notes
We start with red at 0 and end with red at 360. We have drawn 360 rectangles (thickness of 1) and coloured (filled) each one with a different colour. We needed noStroke() otherwise it would be just black! It gives us the

Challenge
You could try to create a circular colour palette.

Code Explanation

`for (let i = 0; i < 360; i++)`	Iterates from 0 to 360 one at a time
`fill(i, 100, 100)`	Increases the hue value one at a time
`rect(i, 1, 1, 400)`	Draws a rectangle incrementing each rectangle one pixel at a time

Figure B5.2


p5*
File
Edit
Sketch
Help
English
Hello, TheHappyCoder!
Auto-refresh
Snippets
by TheHappyCoder
sketch.js
Saved: just now
Preview
function setup()
{
createCanvas(360, 400)
colorMode(HSB)
noStroke()
}
function draw()
{
for (let i = 0; i < 360; i++)
{
fill(i, 100, 100)
rect(i, 1, 1, 400)
}
}
Console
Clear

Sketch B5.3 circle of colour

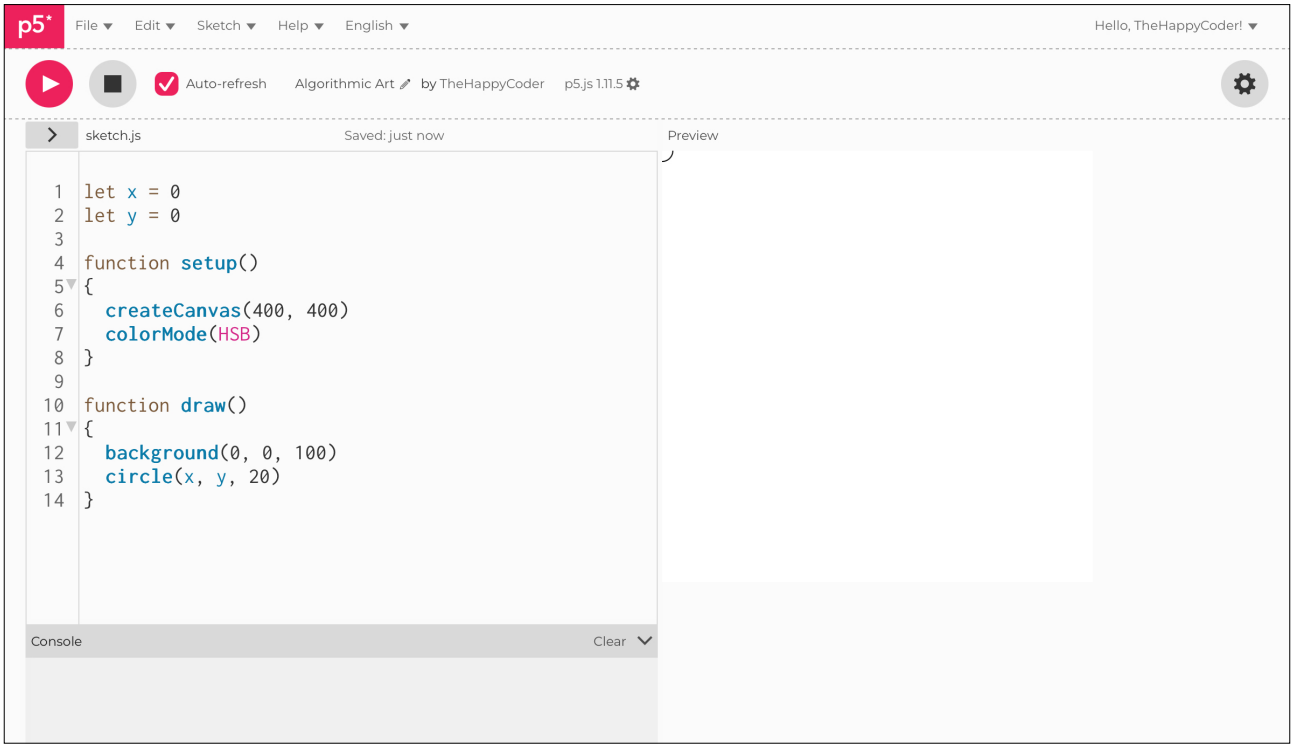
❗ another new sketch.
To get white, you put the saturation to zero.

```
let x = 0
let y = 0

function setup()
{
  createCanvas(400, 400)
  colorMode(HSB)
}

function draw()
{
  background(0, 0, 100)
  circle(x, y, 20)
}
```

🗒️ Notes

Notice how we have given the background a white colour and a circle hiding in the top left hand corner.

Figure B5.3

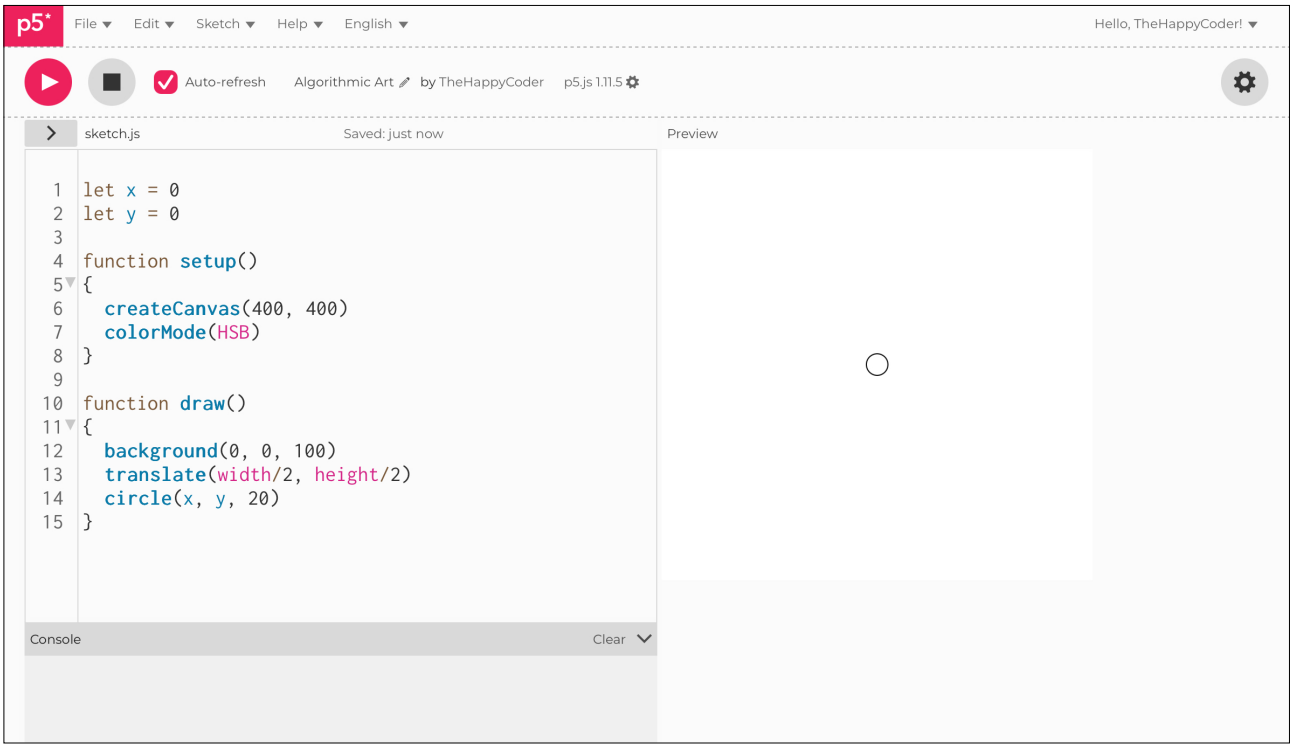

p5*
File
Edit
Sketch
Help
English
Hello, TheHappyCoder!
Auto-refresh
Algorithmic Art
by TheHappyCoder
p5.js 1.11.5
sketch.js
Saved: just now
Preview
let x = 0
let y = 0
function setup()
{
createCanvas(400, 400)
colorMode(HSB)
}
function draw()
{
background(0, 0, 100)
circle(x, y, 20)
}
Console
Clear

Sketch B5.4 centring the circle

It is hard to see, so we will translate the circle to the centre.

```
let x = 0
let y = 0

function setup()
{
  createCanvas(400, 400)
  colorMode(HSB)
}

function draw()
{
  background(0, 0, 100)
  translate(width/2, height/2)
  circle(x, y, 20)
}
```

Figure B5.4



Sketch B5.5 angle of attack

We want to draw a series of small circles around a larger circle, spaced out evenly. We will use a `for()` loop to increment `8°` from `0°` to `360°`. We will do this in `degrees` using `angleMode(DEGREES)`.

```
let x = 0
let y = 0

function setup()
{
  createCanvas(400, 400)
  colorMode(HSB)
  angleMode(DEGREES)
}

function draw()
{
  background(0, 0, 100)
  translate(width/2, height/2)
  for (let angle = 0; angle < 360; angle += 8)
  {
    fill(angle, 100, 100)
    circle(x, y, 20)
  }
}
```

🗒️ Notes

Notice all this seems to do is fill the circle red; this is because the `360°` (and `0°`) value of Hue is red. What happens is that it spins through all the colours very fast in the `for()` loop.

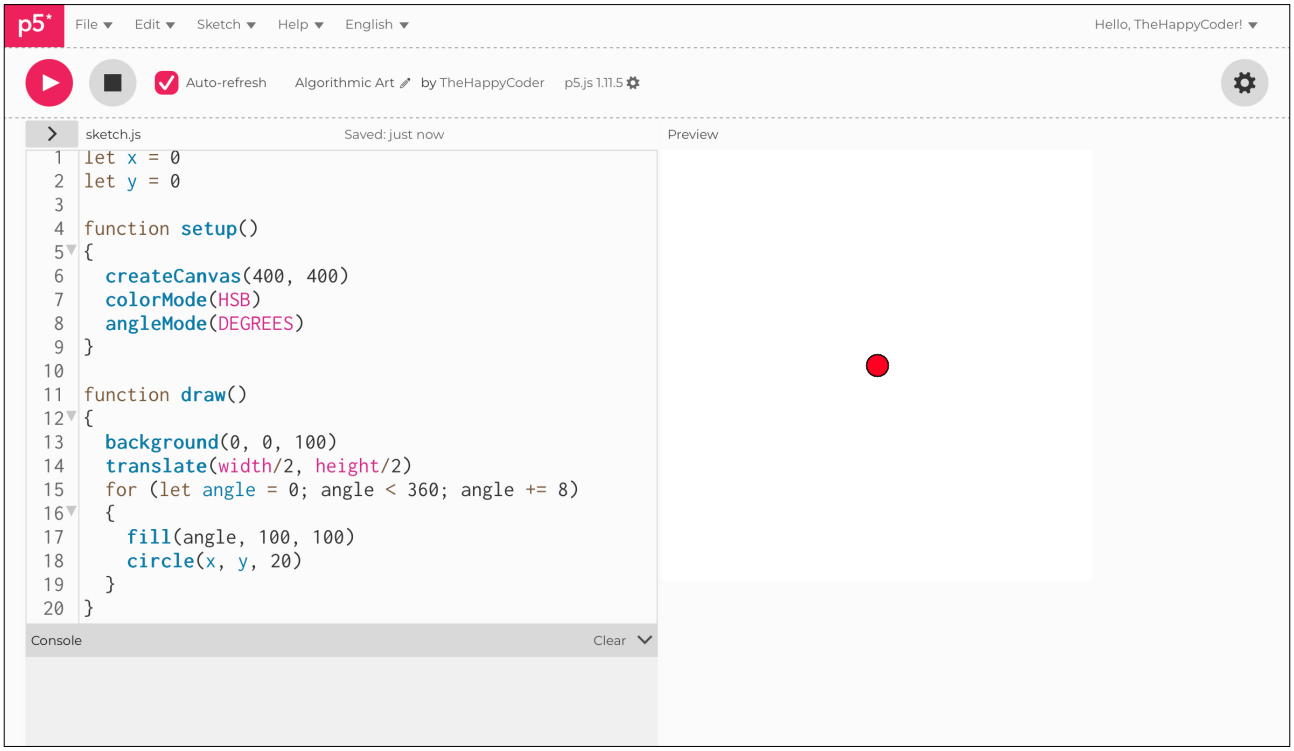
🌻 Challenge

Change the `360` to `180`.

🛠️ Code Explanation

`for (let angle = 0; angle < 360; angle += 8)`	Spin through all the angles from 0°-360° by 8° at a time
`fill(angle, 100, 100)`	Fill the hue value from the angle

Figure B5.5


p5*
File
Edit
Sketch
Help
English
Hello, TheHappyCoder!
Auto-refresh
Algorithmic Art
by TheHappyCoder
p5.js 1.11.5
sketch.js
Saved: just now
Preview
let x = 0
let y = 0
function setup()
{
createCanvas(400, 400)
colorMode(HSB)
angleMode(DEGREES)
}
function draw()
{
background(0, 0, 100)
translate(width/2, height/2)
for (let angle = 0; angle < 360; angle += 8)
{
fill(angle, 100, 100)
circle(x, y, 20)
}
}
Console
Clear

Sketch B5.6 sine and cosine

We use the principles discussed a few units back, where the coordinates on a circle can be described using `sine` and `cosine` for `x` and `y` about the centre of the circle.

```
let x = 0
let y = 0
let radius = 180

function setup()
{
  createCanvas(400, 400)
  colorMode(HSB)
  angleMode(DEGREES)
}

function draw()
{
  background(0, 0, 100)
  translate(width/2, height/2)
  for (let angle = 0; angle < 360; angle += 8)
  {
    fill(angle, 100, 100)
    x = radius * sin(angle)
    y = radius * cos(angle)
    circle(x, y, 20)
  }
}
```

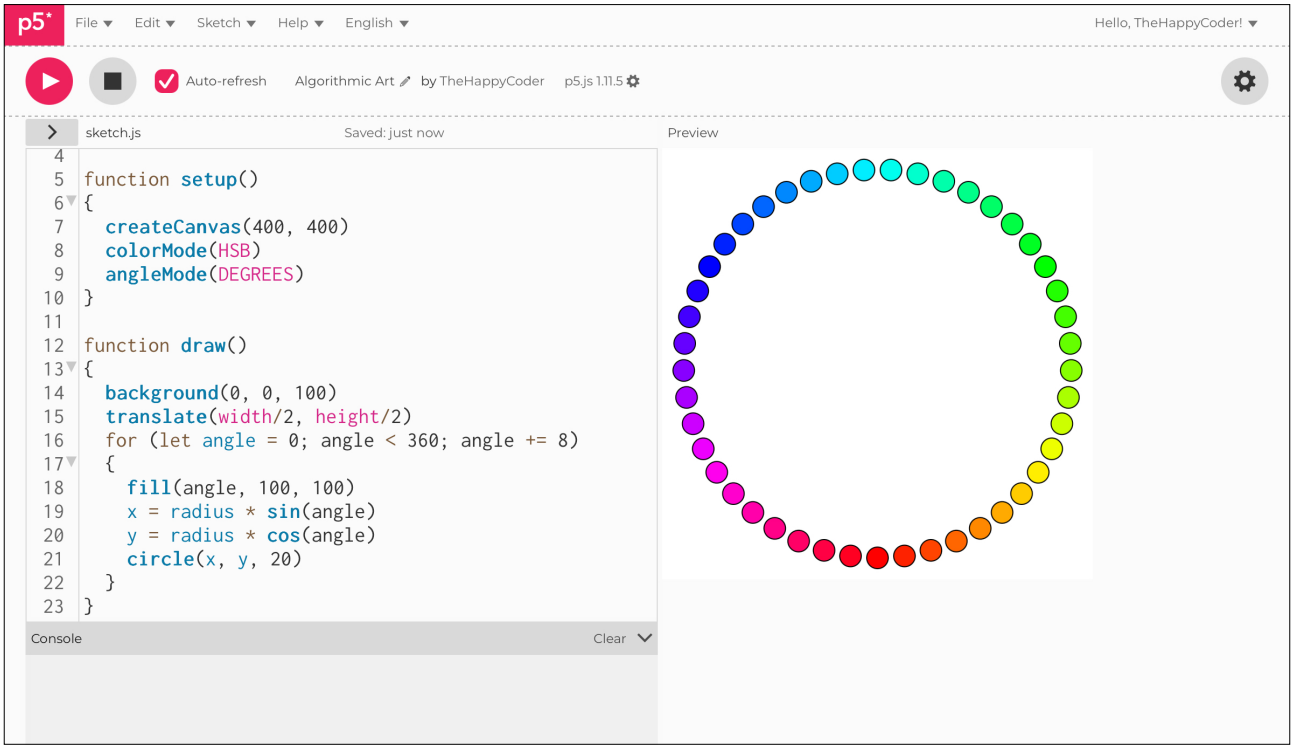
🗒️ Notes

You now have the 360-degree series of circles.

🌻 Challenge

Try different radii and spacings of the circles.

Figure B5.6


p5*
File
Edit
Sketch
Help
English
Hello, TheHappyCoder!
Auto-refresh
Algorithmic Art
by TheHappyCoder
p5.js 1.11.5
sketch.js
Saved: just now
Preview
4
5 function setup()
6 {
7 createCanvas(400, 400)
8 colorMode(HSB)
9 angleMode(DEGREES)
10 }
11
12 function draw()
13 {
14 background(0, 0, 100)
15 translate(width/2, height/2)
16 for (let angle = 0; angle < 360; angle += 8)
17 {
18 fill(angle, 100, 100)
19 x = radius * sin(angle)
20 y = radius * cos(angle)
21 circle(x, y, 20)
22 }
23 }
Console
Clear

Sketch B5.7 one final thing

Let's draw a nice box round the canvas so we can see where the canvas starts. Even for something as simple as this, you need to think about translation and the impact on coordinates.

```
let x = 0
let y = 0
let radius = 180

function setup()
{
  createCanvas(400, 400)
  colorMode(HSB)
  angleMode(DEGREES)
}

function draw()
{
  background(0, 0, 100)
  translate(width/2, height/2)
  for (let angle = 0; angle < 360; angle += 8)
  {
    fill(angle, 100, 100)
    x = radius * sin(angle)
    y = radius * cos(angle)
    circle(x, y, 20)
  }
  noFill()
  rect(-width/2, -height/2, width, height)
}
```

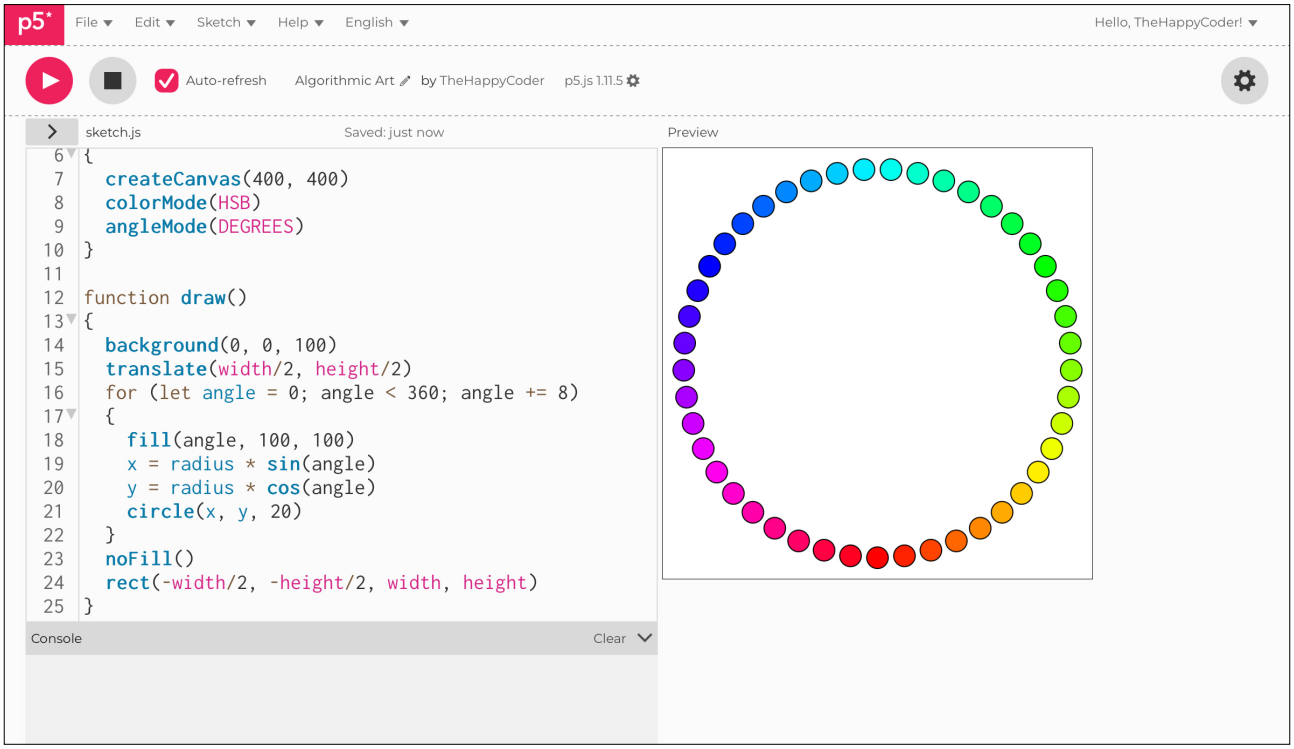
🗒️ Notes

Looks a lot nicer.

🌻 Challenge

If you put it inside the loop, what would you need to change?

Figure B5.7


p5*
File
Edit
Sketch
Help
English
Hello, TheHappyCoder!
Auto-refresh
Algorithmic Art
by TheHappyCoder
p5.js 1.11.5
sketch.js
Saved: just now
Preview
{
createCanvas(400, 400)
colorMode(HSB)
angleMode(DEGREES)
}
function draw()
{
background(0, 0, 100)
translate(width/2, height/2)
for (let angle = 0; angle < 360; angle += 8)
{
fill(angle, 100, 100)
x = radius * sin(angle)
y = radius * cos(angle)
circle(x, y, 20)
}
noFill()
rect(-width/2, -height/2, width, height)
}
Console
Clear

Sketch B5.8 100 rectangles

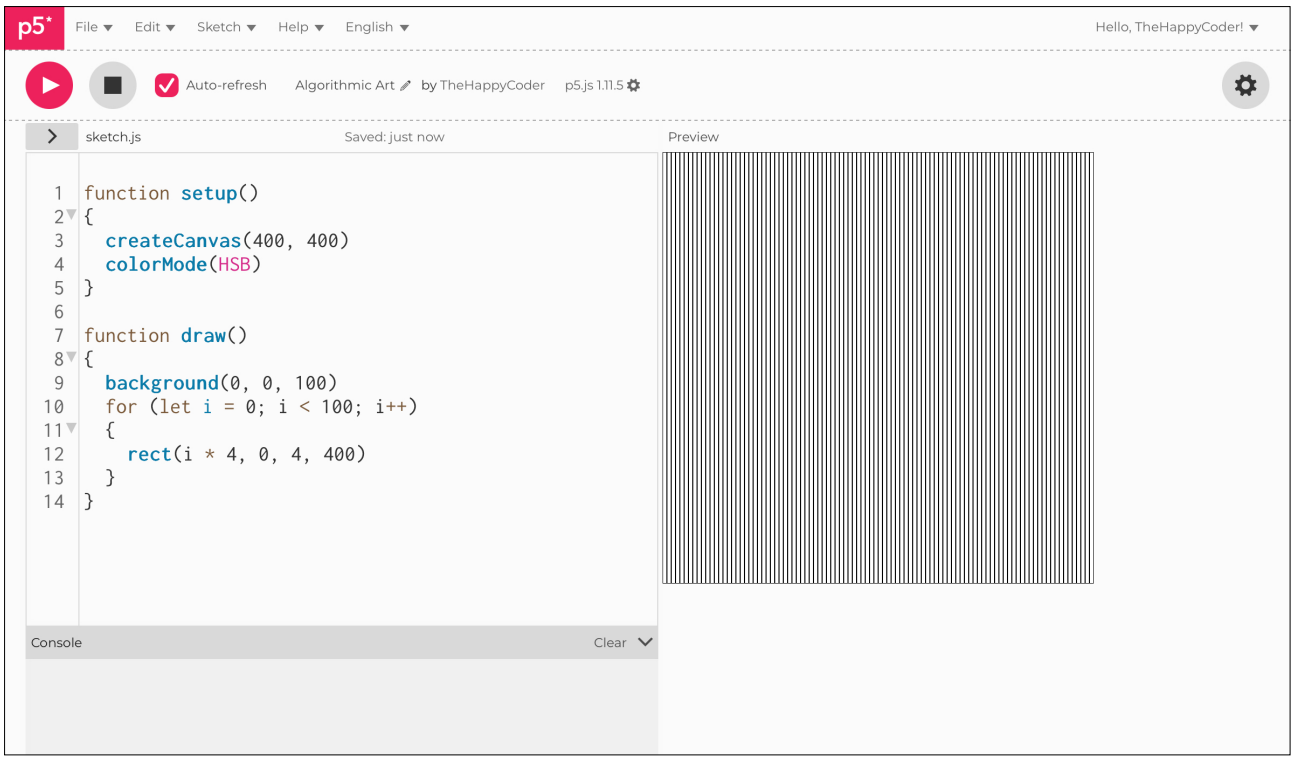
❗ Starting a new sketch where we loop through 100 rectangles, with each rectangle being 4 pixels wide and 400 pixels long; hence, we multiply the i variable by 4.

```
function setup()
{
  createCanvas(400, 400)
  colorMode(HSB)
}

function draw()
{
  background(0, 0, 100)
  for (let i = 0; i < 100; i++)
  {
    rect(i * 4, 0, 4, 400)
  }
}
```

🗒️ Notes

We just get a bunch of rectangles from left to right.

Figure B5.8


p5*
File
Edit
Sketch
Help
English
Hello, TheHappyCoder!
Auto-refresh
Algorithmic Art
by TheHappyCoder
p5.js 1.11.5
sketch.js
Saved: just now
Preview
function setup()
{
createCanvas(400, 400)
colorMode(HSB)
}
function draw()
{
background(0, 0, 100)
for (let i = 0; i < 100; i++)
{
rect(i * 4, 0, 4, 400)
}
}
Console
Clear

Sketch B5.9 HSB saturation

Then we fill each rectangle with a hue of 0, which is red; the saturation (second argument) we increase by 1 on each iteration of the loop, keeping the brightness (third argument) at maximum (100).

```
function setup()
{
  createCanvas(400, 400)
  colorMode(HSB)
  noStroke()
}

function draw()
{
  background(0, 0, 100)
  for (let i = 0; i < 100; i++)
  {
    fill(0, i, 100)
    rect(i * 4, 0, 4, 400)
  }
}
```

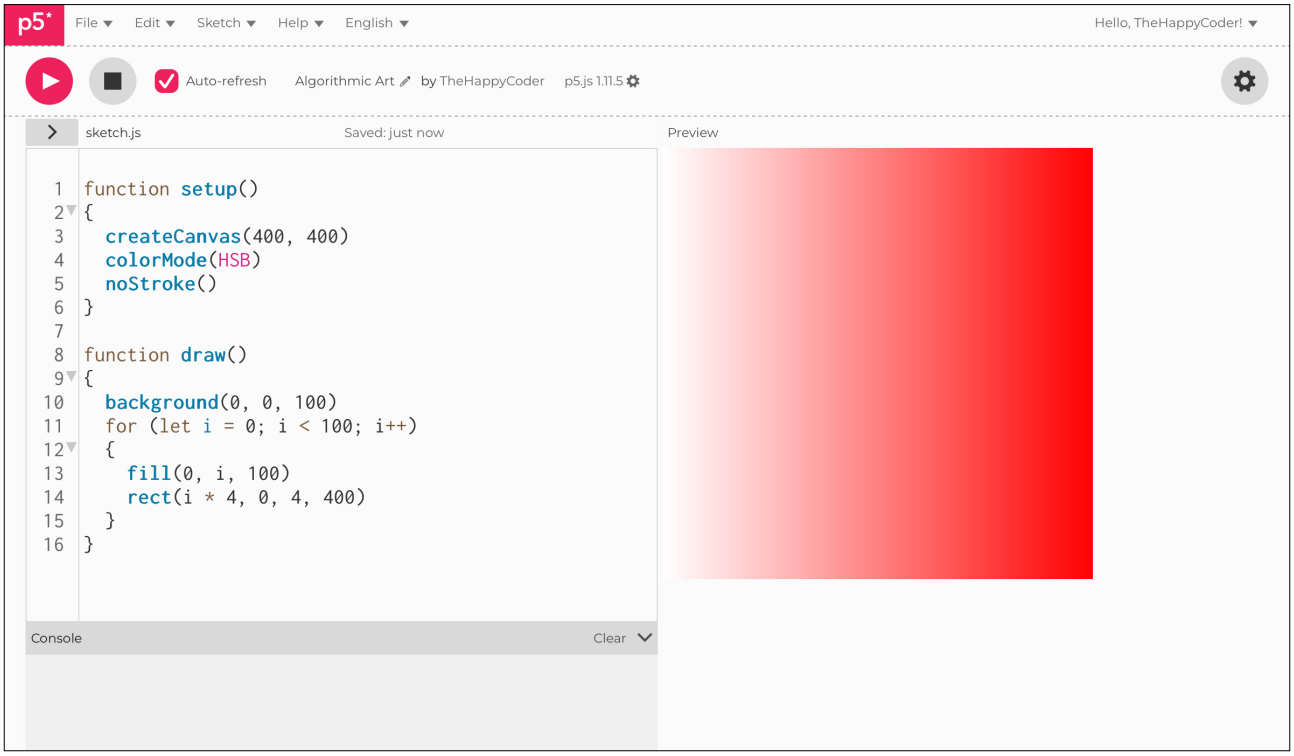
🗒️ Notes

A saturation of 0 is no saturation (white), and 100 is the maximum; then everything in between.

🌻 Challenge

Do this with other hues.

Figure B5.9


p5*
File
Edit
Sketch
Help
English
Hello, TheHappyCoder!
Auto-refresh
Algorithmic Art
by TheHappyCoder
p5.js 1.11.5
sketch.js
Saved: just now
Preview
1 function setup()
2 {
3 createCanvas(400, 400)
4 colorMode(HSB)
5 noStroke()
6 }
7
8 function draw()
9 {
10 background(0, 0, 100)
11 for (let i = 0; i < 100; i++)
12 {
13 fill(0, i, 100)
14 rect(i * 4, 0, 4, 400)
15 }
16 }
Console
Clear

Sketch B5.10 HSB brightness

Now for the brightness, this is the third argument: we change the `fill()` function, giving it `0` hue, `100` saturation, and an incrementing `i` variable for the brightness. This means that it starts off with `zero` brightness and increments to `full` brightness (100%).

```
function setup()
{
  createCanvas(400, 400)
  colorMode(HSB)
  noStroke()
}

function draw()
{
  background(0, 0, 100)
  for (let i = 0; i < 100; i++)
  {
    fill(0, 100, i)
    rect(i * 4, 0, 4, 400)
  }
}
```

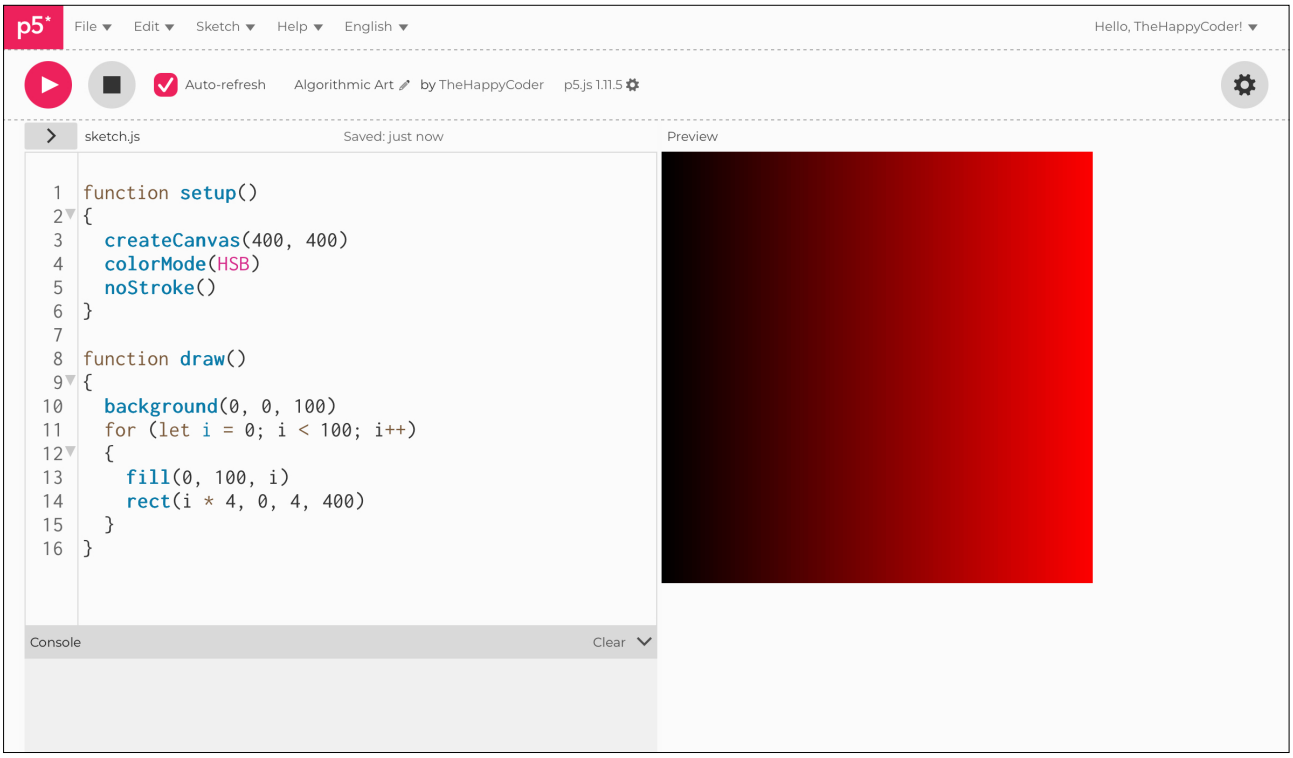
Notes

Fairly obvious results.

Challenges

1. Create a palette of colours that you might want to use regularly.
2. Use sliders for Hue, saturation and Brightness.

Figure B5.10


p5*
File
Edit
Sketch
Help
English
Hello, TheHappyCoder!
Auto-refresh
Algorithmic Art
by TheHappyCoder
p5.js 1.11.5
sketch.js
Saved: just now
Preview
function setup()
{
createCanvas(400, 400)
colorMode(HSB)
noStroke()
}
function draw()
{
background(0, 0, 100)
for (let i = 0; i < 100; i++)
{
fill(0, 100, i)
rect(i * 4, 0, 4, 400)
}
}
Console
Clear