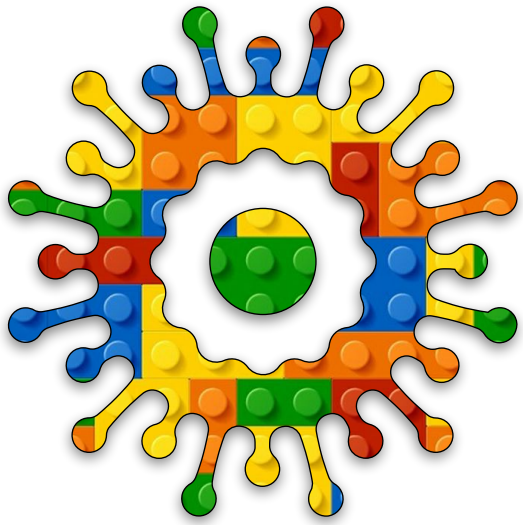
Algorithmic Art

Module A

Unit #1

Noise



Content

Module A Unit #1 Noise

Introduction to perlin noise

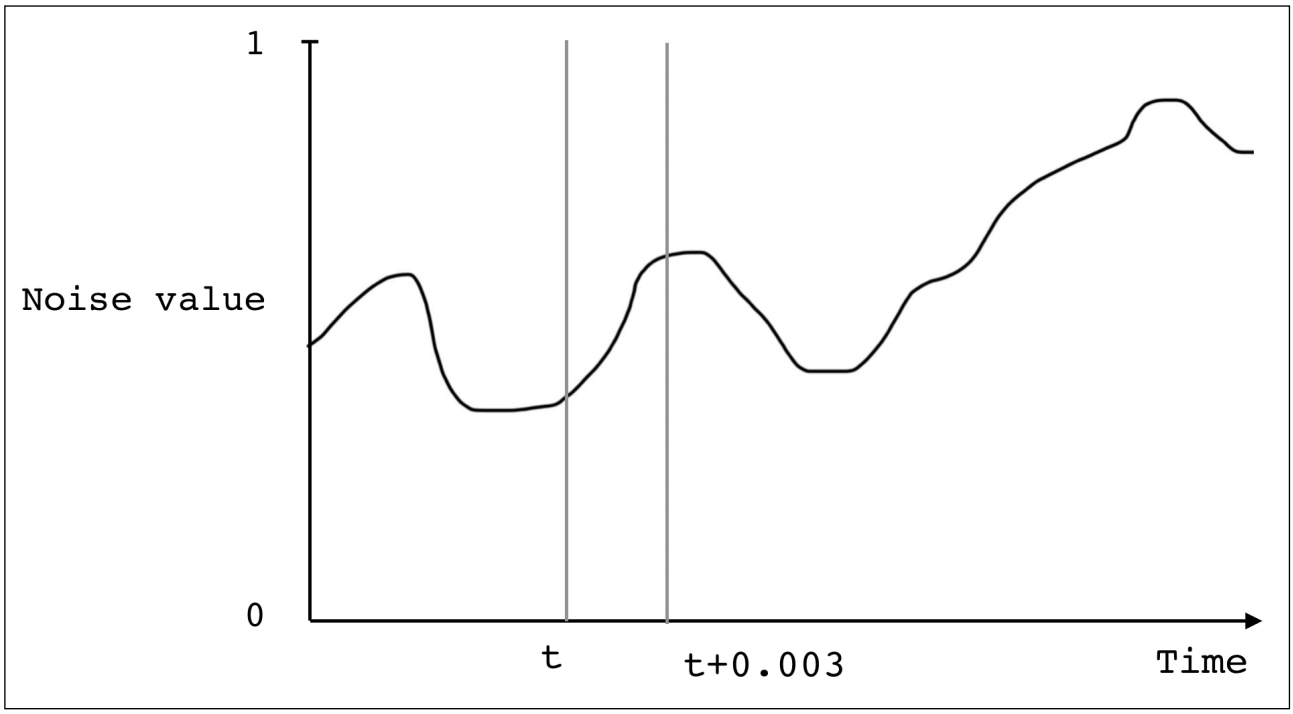
Perlin `noise()` returns a random value between `0.0` and `1.0` at a specific point in time. You specify the start time. It is then incremented along this smooth random time line in steps; the smaller the steps (for instance, `0.005`) means there are very small steps but the changes are therefore smoother, whereas larger steps (`0.03`) obviously create a much greater degree of randomness and possibly less smooth in the changes. It seems to work best between `0.005` and `0.03`.

If this seems strange at first, it is because it is less intuitive than just plucking a random number out of the air. If you have two variables that you want to have different random noise outcomes, you simply use the `noise()` function but start at different times (points along the line), for instance, one variable could start at `3` and the other starts at `100`.

The beauty of this is that it is random but it bears some relationship with the previous random number at a particular point in time. So if you move it on a small increment, you get a slight adjustment to the random value.

If you want to understand how it works, there are a number of articles around; the `Nature of Code` is a good reference point.

Figure 1: perlin noise graph



Sketch A1.1 starting with a standard sketch

Starting with our normal basic sketch.

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

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

Sketch A1.2 randomly moving circle

We start the circle in the centre of the canvas and randomly move it. We are just going to use the `random()` function to move the circle around the canvas. You will notice that it is not very smooth or fluid; `noise` will give you something more natural.

```
let x = 200
let y = 200

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

function draw()
{
  background(220)
  circle(x, y, 200)
  x = x + random(-2, 2)
  y = y + random(-2, 2)
}
```

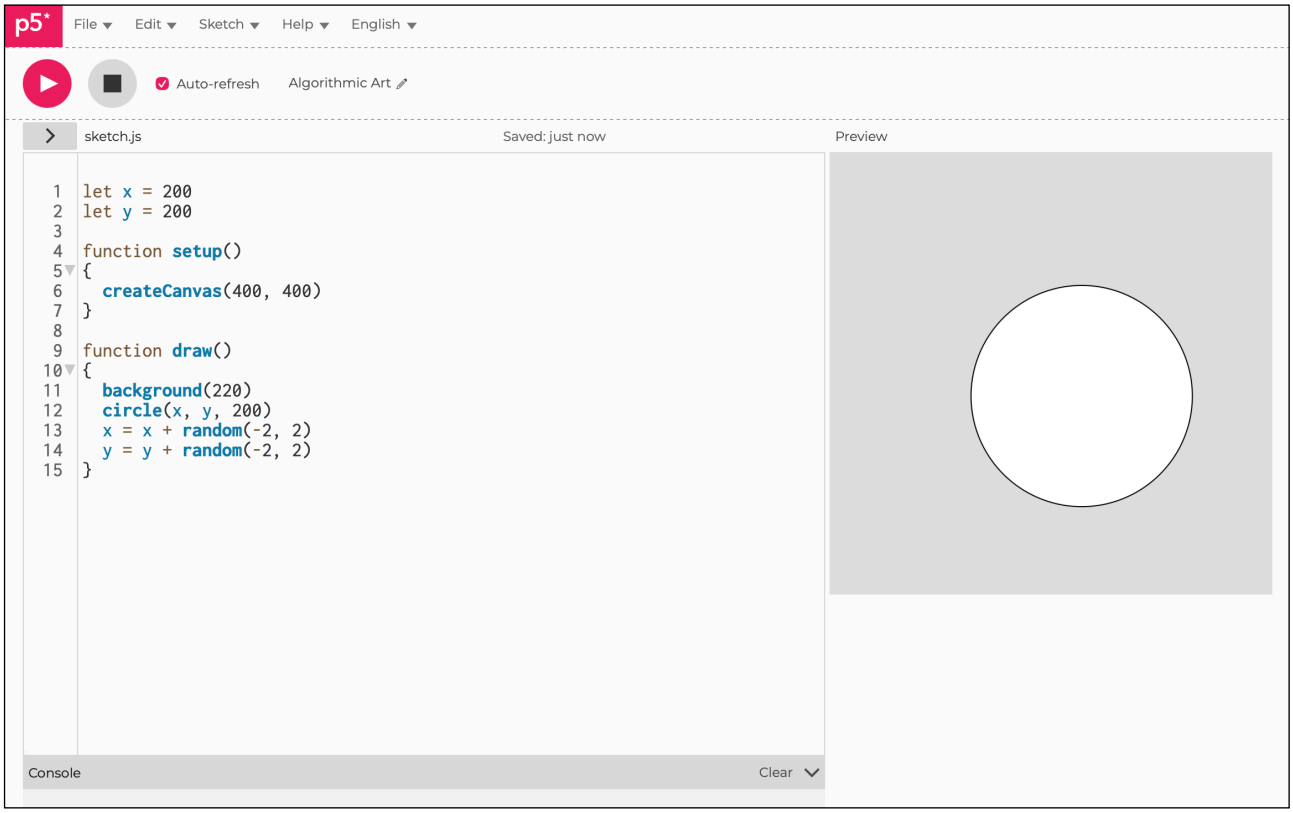
Challenges

1. Try random values of (5, -5).
2. Try different values for x and for y.

Code Explanation

`x = x + random(-2, 2)`	Add a random value between -2 and 2 to the x value on each iteration
`y = y + random(-2, 2)`	Add a random value between -2 and 2 to the y value on each iteration

Figure A1.2


p5*
File
Edit
Sketch
Help
English
Auto-refresh
Algorithmic Art
sketch.js
Saved: just now
Preview
let x = 200
let y = 200
function setup()
{
createCanvas(400, 400)
}
function draw()
{
background(220)
circle(x, y, 200)
x = x + random(-2, 2)
y = y + random(-2, 2)
}
Console
Clear

Sketch A1.3 smooth random movement

We have replaced the `random()` function with the `noise()` function. Notice that the jerkiness has gone, replaced by a much smoother movement, almost as if floating in the air. Also, it looks a lot more complicated. We have two start times (`3` and `10`). Because `noise` returns values between `0` and `1`, we use the `map()` function to scale the movement up to the `width` and the `height` of the canvas. Then, on each iteration, we move along the `noise` time line by `0.005` increments.

```
let timeX = 3
let timeY = 10

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

function draw()
{
  background(220)
  let x = map(noise(timeX), 0, 1, 0, width)
  let y = map(noise(timeY), 0, 1, 0, height)
  circle(x, y, 200)
  timeX = timeX + 0.005
  timeY = timeY + 0.005
}
```

Notes

To see how noise works and why it is so much better than just `random()`, this short programme illustrates the smoothness of the movement.

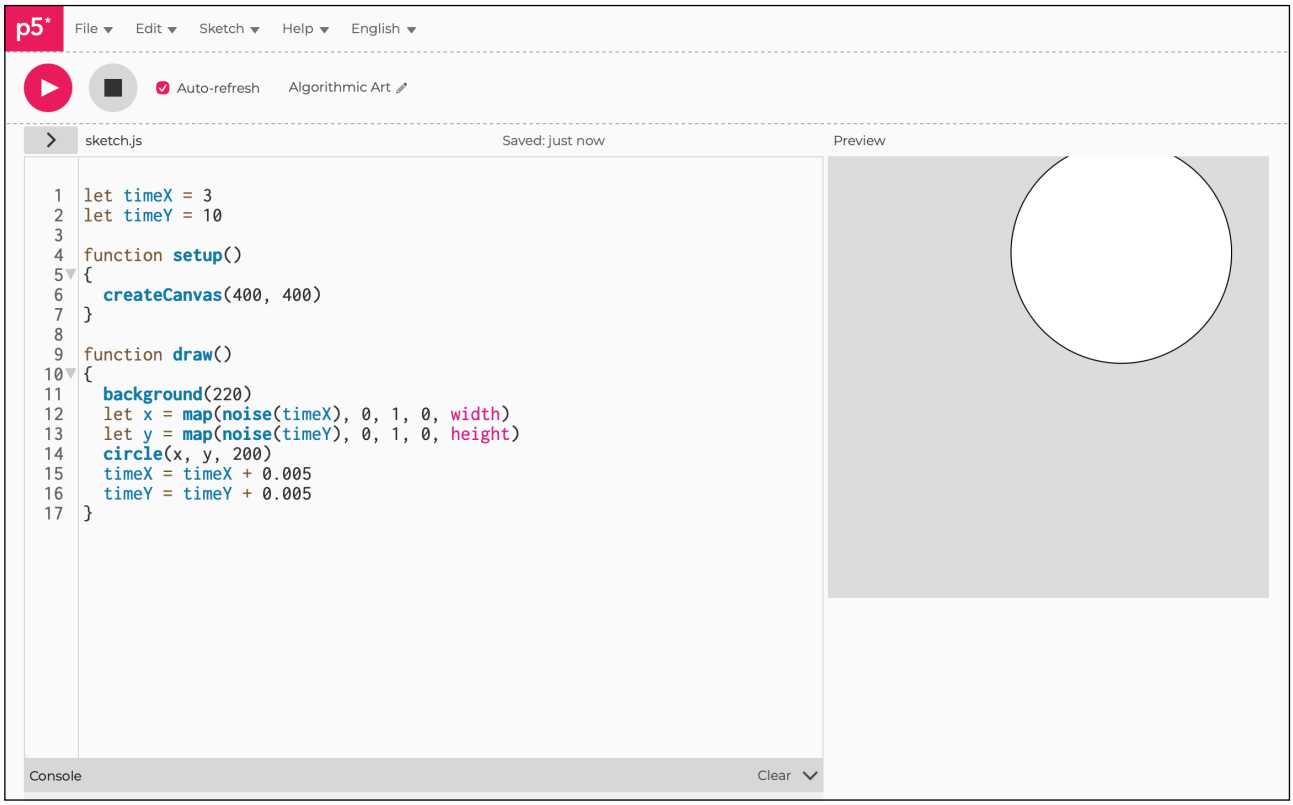
🌻 Challenges

1. Try different increments for timeX and timeY.
2. What happens if you give them both the same start on the timeline?
3. Try: `timeX = timeX + 0.05`.
4. Replace it with `timeX += 0.005`.

🛠️ Code Explanation

`let timeX = 3`	The starting value for the x timeline
`let timeY = 10`	The starting value for the y timeline
`let x = map(noise(timeX), 0, 1, 0, width)`	Maps the timeX value to the width of the canvas
`let y = map(noise(timeY), 0, 1, 0, height)`	Maps the timeY value to the height of the canvas
`timeX = timeX + 0.005`	Adds an increment to the timeX timeline
`timeY = timeY + 0.005`	Adds an increment to the timeY timeline

Figure A1.3


p5*
File
Edit
Sketch
Help
English
Auto-refresh
Algorithmic Art
sketch.js
Saved: just now
Preview
let timeX = 3
let timeY = 10
function setup()
{
createCanvas(400, 400)
}
function draw()
{
background(220)
let x = map(noise(timeX), 0, 1, 0, width)
let y = map(noise(timeY), 0, 1, 0, height)
circle(x, y, 200)
timeX = timeX + 0.005
timeY = timeY + 0.005
}
Console
Clear

Sketch A1.4 random colour B/W

❗ Start a new sketch.
Adding a colour element to change the grayscale.

```
let timeCol = 3
let col = 0

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

function draw()
{
  background(220)
  fill(col)
  col = map(noise(timeCol), 0, 1, 0, 255)
  circle(width/2, height/2, 200)
  timeCol = timeCol + 0.005
}
```

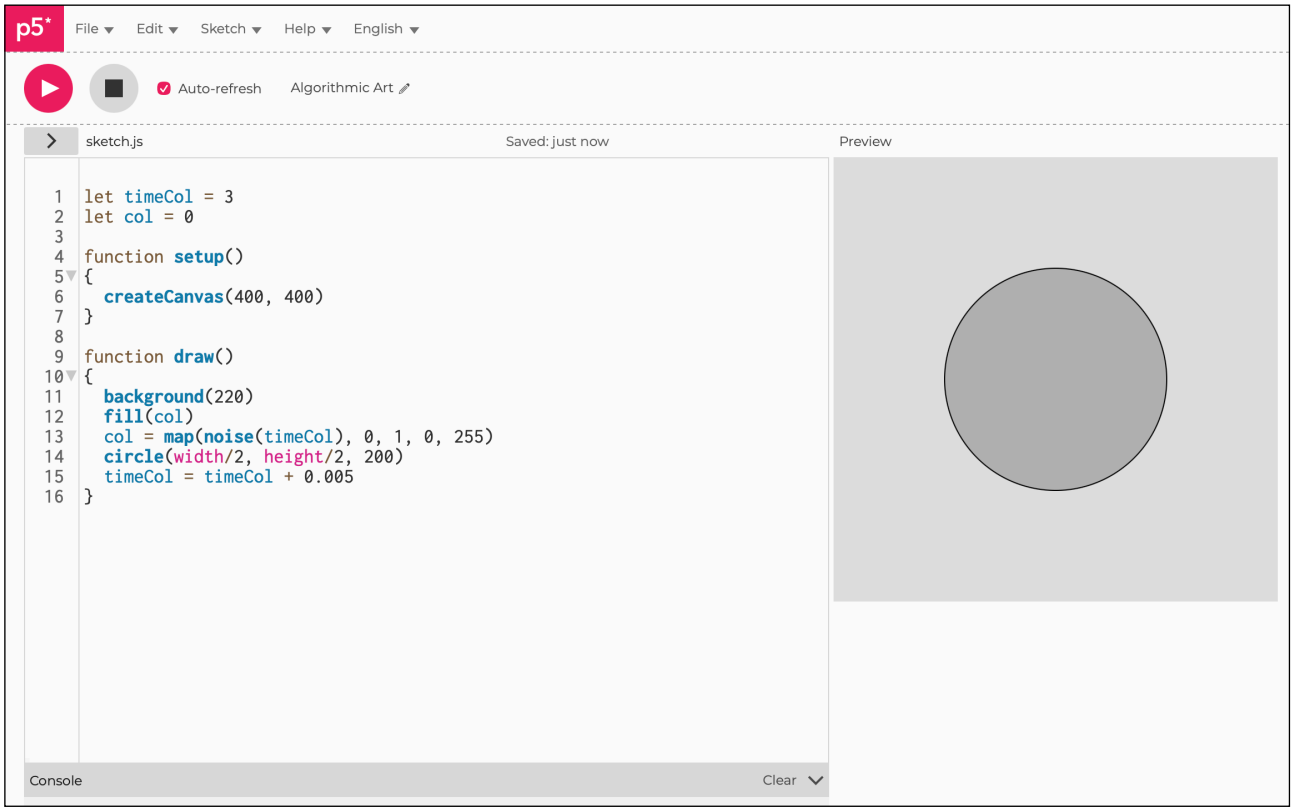
🗒️ Notes

The colour of the circle changes gradually through the greyscale values between `0` and `255`. The transition from one level of grey to another is also smoother and much more pleasing to the eye.

🌻 Challenge

Have the background change as well using `noise()`.

Figure A1.4


p5*
File
Edit
Sketch
Help
English
Auto-refresh
Algorithmic Art
sketch.js
Saved: just now
Preview
let timeCol = 3
let col = 0
function setup()
{
createCanvas(400, 400)
}
function draw()
{
background(220)
fill(col)
col = map(noise(timeCol), 0, 1, 0, 255)
circle(width/2, height/2, 200)
timeCol = timeCol + 0.005
}
Console
Clear

Sketch A1.5 random colour RGB

❗ remove: the non relevant code, we are replacing a single colour with the RGB ones instead.

Changing all three to create a flow of colour changes.

```
let timeRed = 30
let timeGreen = 300
let timeBlue = 3000
let colRed = 0
let colGreen = 0
let colBlue = 0

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

function draw()
{
  background(220)
  fill(colRed, colGreen, colBlue)
  colRed = map(noise(timeRed), 0, 1, 0, 255)
  colGreen = map(noise(timeGreen), 0, 1, 0, 255)
  colBlue = map(noise(timeBlue), 0, 1, 0, 255)
  circle(width/2, height/2, 200)
  timeRed += 0.003
  timeGreen += 0.003
  timeBlue += 0.003
}
```

🗒️ Notes

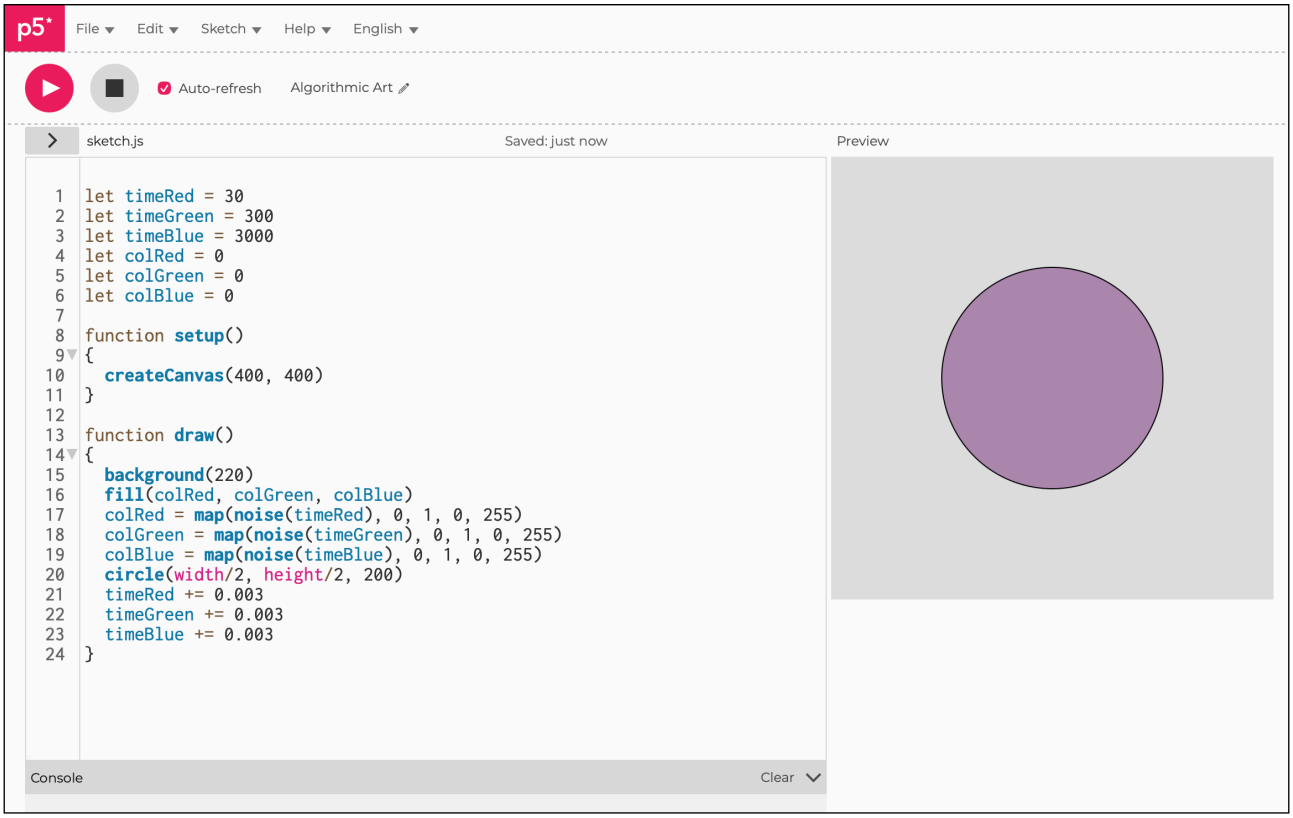
Like the previous sketch, we get a smoother and more subtle transition from one colour to the next.

This can take a little while to get your head around. With random, you give it a number to randomise up to and from, but with noise, you are picking a point on a random timeline. Play with this and persevere; it will become a bit more intuitive.

🌻 Challenges

1. Change the background colour, easy hint: just swap the colour variables round, for instance: `background(colGreen, colBlue, colRed)`
2. Movement, colour and size all at the same time.

Figure A1.5


p5*
File
Edit
Sketch
Help
English
Auto-refresh
Algorithmic Art
sketch.js
Saved: just now
Preview
let timeRed = 30
let timeGreen = 300
let timeBlue = 3000
let colRed = 0
let colGreen = 0
let colBlue = 0
function setup()
{
createCanvas(400, 400)
}
function draw()
{
background(220)
fill(colRed, colGreen, colBlue)
colRed = map(noise(timeRed), 0, 1, 0, 255)
colGreen = map(noise(timeGreen), 0, 1, 0, 255)
colBlue = map(noise(timeBlue), 0, 1, 0, 255)
circle(width/2, height/2, 200)
timeRed += 0.003
timeGreen += 0.003
timeBlue += 0.003
}
Console
Clear