Section A
Unit #8
Noise

Content

Section A
Unit #8 Noise

Unit #8 noise

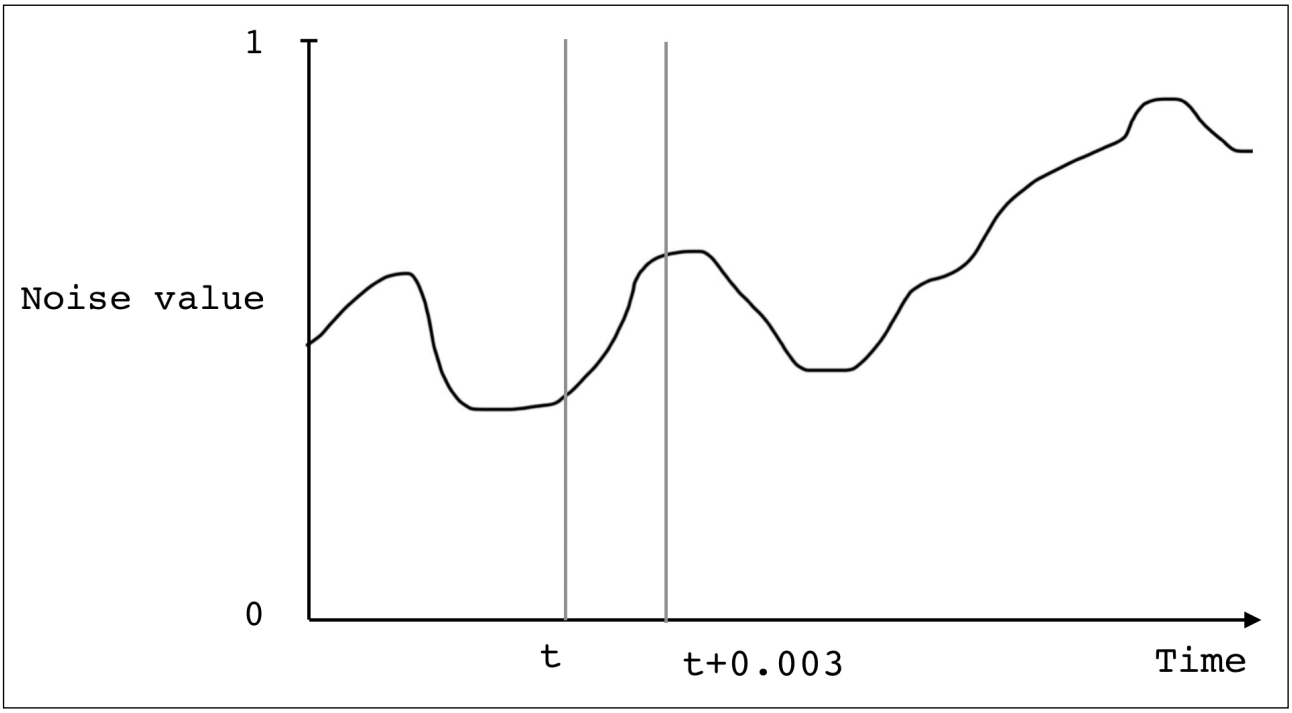
Perlin **noise()** returns a random value between 0.0 and 1.0 at a specific time. You specify the time. It is increment in steps along the time line so small steps work best e.g. 0.005-0.03.

If this seems strange then don't let it be. If you have two variables that you want to have a different random outcome you simply grab the **noise()** at different times, e.g. 3 or 3000.

The beauty of this is that it is random but it has 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 really want to understand how it works and who invented it (and why!) then I suggest you read up about it in Dan Shiffman's book/website 'Nature of Code' which is currently just for the processing language.

Perlin noise graph



Sketch A8.1 randomly moving circle

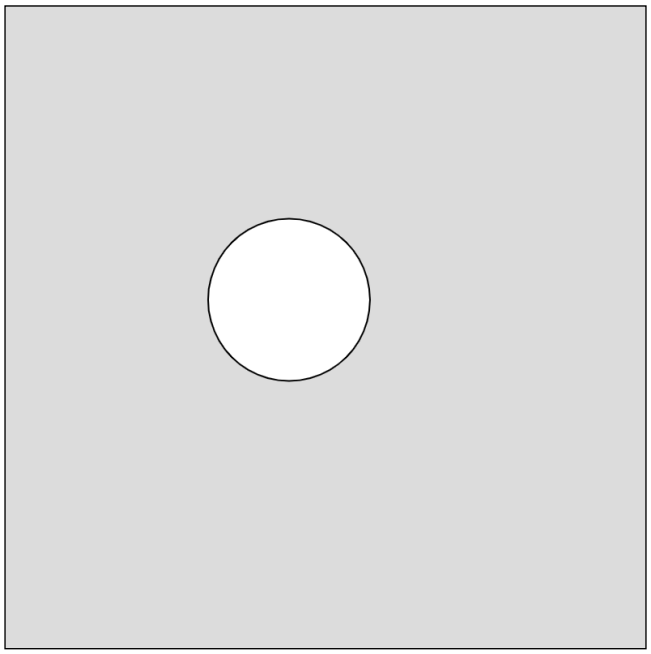
We are just going to use the random() function to move a circle around the canvas. You will notice that it is quite jerky, especially if you increase the randomness from 2 to say 5

```
let x = 200
let y = 200

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

function draw()
{
  background(220)
  circle(x, y, 100)
  x = x + random(-2, 2)
  y = y + random(-2, 2)
}
```

Jerky movement, not smooth



Sketch A8.2 randomly moving circle

What random noise movement looks like. Notice that the jerkiness has gone replaced by a smooth movement as it seems to float in the air

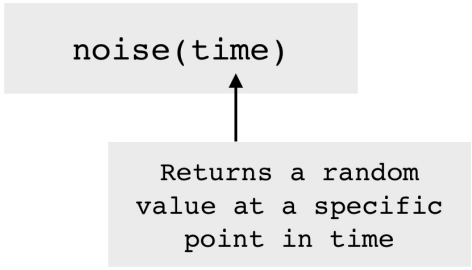
```
let timeX = 3
let timeY = 10

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, 100)
  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. Try replacing noise(time) with random(time) and see the difference. Play with the time variable and the increment (0.005). Mapping is used because noise() returns a value between 0 and 1, so you need to scale it up accordingly



Challenge

1. Try different increments for timeX and timeY
2. Try: time = time + 0.05

Sketch A8.3 random colour B/W

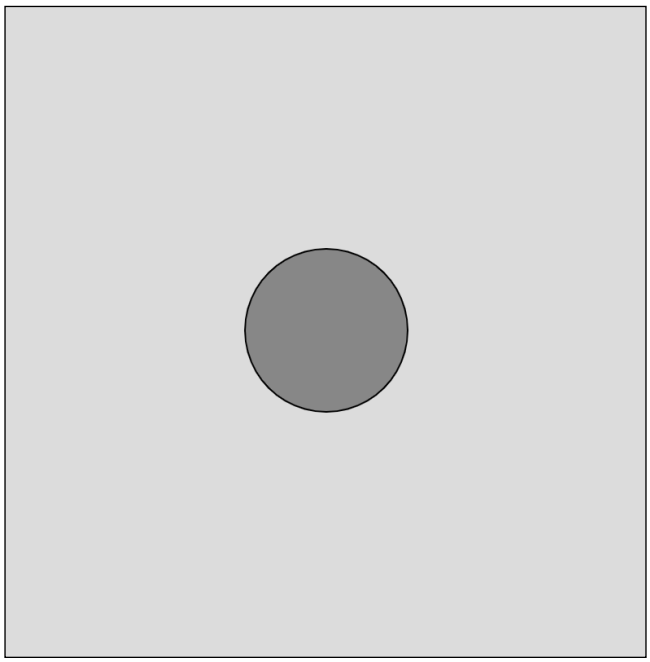
Start a new sketch. Adding colour element to change the greyscale

```
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, 100)
  timeCol = timeCol + 0.005
}
```

Colour of the circle changes gradually through the greyscale values between 0 and 255



Notes

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()

Sketch A8.4 random colour RGB

Changing all three to create a flow of colour change

```
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.

Challenge

Change the background colour, easy hint: just swap the colour variables round eg `background(colGreen, colBlue, colRed)`

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 time line. Play with this and persevere it will become a bit more intuitive.