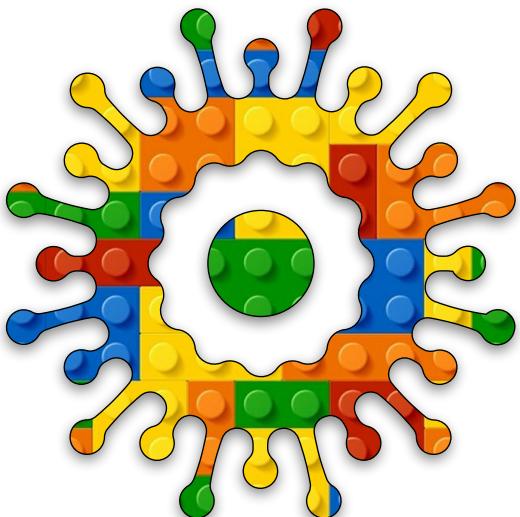


Creative Coding

Module B

Unit #3

text





Module B Unit #3 text

- Sketch B3.1 starting sketch
- Sketch B3.2 text size
- Sketch B3.3 aligning the text
- Sketch B3.4 colour the text
- Sketch B3.5 border colour
- Sketch B3.6 stroke
- Sketch B3.7 translate
- Sketch B3.8 angle of degree
- Sketch B3.9 rotating
- Sketch B3.10 mouse mover
- Sketch B3.11 not a string
- Sketch B3.12 and mouseY
- Sketch B3.13 static
- Sketch B3.14 adding a string



Introduction to using text

As well as shapes, you can use text, either directly as a string or as a variable value. This gives you many opportunities to manipulate text (words, letters or numbers) or values (numbers) for various possibilities. Although you get one font with the basic sketch, you can upload lots of other fonts (for another unit later). For now, we will stick with the default font.



Sketch B3.1 starting sketch

We have a line of code that uses the `text()` function. The actual text is in speech marks (single or double), this is a string. The following two arguments are the distance from the left-hand side (`x`) and the distance from the top edge (`y`).

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

function draw()
{
    background(220)
    text('Hello', 200, 200)
}
```

Notes

You get to print some text on the canvas; the default size is quite small, but we can change that.

Challenge

Type some other words.

Code Explanation

<code>text('Hello', 200, 200)</code>	Write the word 'Hello' 200 from the left and 200 from the top
--------------------------------------	--

Figure B3.1

The screenshot shows the p5.js IDE interface. At the top, there's a toolbar with icons for play, stop, and refresh, followed by menu items: File, Edit, Sketch, Help, and English. To the right of the menu is a status bar displaying "Hello, TheHappyCoder!". The main workspace is divided into two sections: "sketch.js" on the left and "Preview" on the right. In the "sketch.js" section, the following code is written:

```
1 function setup()
2 {
3     createCanvas(400, 400)
4 }
5
6 function draw()
7 {
8     background(220)
9     text('Hello', 200, 200)
10}
```

In the "Preview" section, the word "Hello" is displayed in a simple black font at the center of a 400x400 pixel canvas.



Sketch B3.2 text size

You will notice that it is quite small; the default size is **12** pixels. We can make it bigger than that using the function **textSize()** and specifying the size of the font.

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

function draw()
{
    background(220)
    textSize(36)
    text('Hello', 200, 200)
}
```

Notes

Much better.

Code Explanation

textSize(36)	Specifies the size of the text
--------------	--------------------------------

Figure B3.2

The screenshot shows the p5.js web-based IDE interface. At the top, there's a navigation bar with 'File ▾', 'Edit ▾', 'Sketch ▾', 'Help ▾', and 'English ▾'. On the right side of the bar, it says 'Hello, TheHappyCoder! ▾'. Below the bar, there are icons for play/pause, stop, auto-refresh (with a checked checkbox), and a gear for settings. The main workspace is divided into two sections: 'sketch.js' on the left and 'Preview' on the right. In the 'sketch.js' section, the code is displayed:

```
1 function setup()
2 {
3   createCanvas(400, 400)
4 }
5
6 function draw()
7 {
8   background(220)
9   textSize(36)
10  text('Hello', 200, 200)
11 }
```

The 'Preview' section shows a gray canvas with the word 'Hello' centered in a large, black, sans-serif font at approximately [200, 200]. Below the preview area, there's a 'Console' tab and a 'Clear ▾' button.



Sketch B3.3 aligning the text

Although we have set the text in the centre of the canvas, it clearly isn't. We can correct that so that the coordinates are specific to the centre of the text, not the top-left-hand corner. We use a function called **textAlign()**. We can tell it where we want the origin to be. In this case, at the very centre of the text, so we use two arguments for the horizontal and vertical alignment.

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

function draw()
{
    background(220)
    textSize(36)
    textAlign(CENTER, CENTER)
    text('Hello', 200, 200)
}
```

Notes

It has shifted the text ever so slightly to the centre of the canvas.

Code Explanation

<code>textAlign(CENTER, CENTER)</code>	Centres the text both vertically and horizontally
--	---

Figure B3.3

The screenshot shows the p5.js code editor interface. At the top, there's a toolbar with icons for play, stop, and refresh, followed by the text "Hello, TheHappyCoder! ▾". Below the toolbar, the file name "sketch.js" is displayed along with the message "Saved: just now". The main area contains the following p5.js code:

```
1 function setup()
2 {
3     createCanvas(400, 400)
4 }
5
6 function draw()
7 {
8     background(220)
9     textSize(36)
10    textAlign(CENTER, CENTER)
11    text('Hello', 200, 200)
12 }
```

To the right of the code editor is a preview window showing a light gray canvas with the word "Hello" centered in a large, black, sans-serif font.



Sketch B3.4 colour the text

We will make the text size even bigger to see the effect more clearly. We will also make the text red.

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

function draw()
{
    background(220)
    textSize(136)
    textAlign(CENTER, CENTER)
    fill('red')
    text('Hello', 200, 200)
}
```

Notes

With smallish text, we get a slight distortion with the colours.

Challenge

Try other colours and alpha.

Figure B3.4

The screenshot shows the p5.js IDE interface. At the top, there's a toolbar with icons for play, stop, and refresh, followed by menu items: File, Edit, Sketch, Help, and English. To the right of the toolbar, it says "Hello, TheHappyCoder! ▾". Below the toolbar, the file name "sketch.js" is displayed along with the message "Saved: just now". A "Preview" button is also present.

The main area contains the following code:

```
1 function setup()
2 {
3     createCanvas(400, 400)
4 }
5
6 function draw()
7 {
8     background(220)
9     textSize(136)
10    textAlign(CENTER, CENTER)
11    fill('red')
12    text('Hello', 200, 200)
13 }
```

On the right side of the interface, there is a preview window showing the word "Hello" in large, bold red font centered on a gray background.



Sketch B3.5 border colour

We can add a border colour using the **stroke()** function.

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

function draw()
{
    background(220)
    textSize(136)
    textAlign(CENTER, CENTER)
    fill('red')
    stroke('blue')
    text('Hello', 200, 200)
}
```

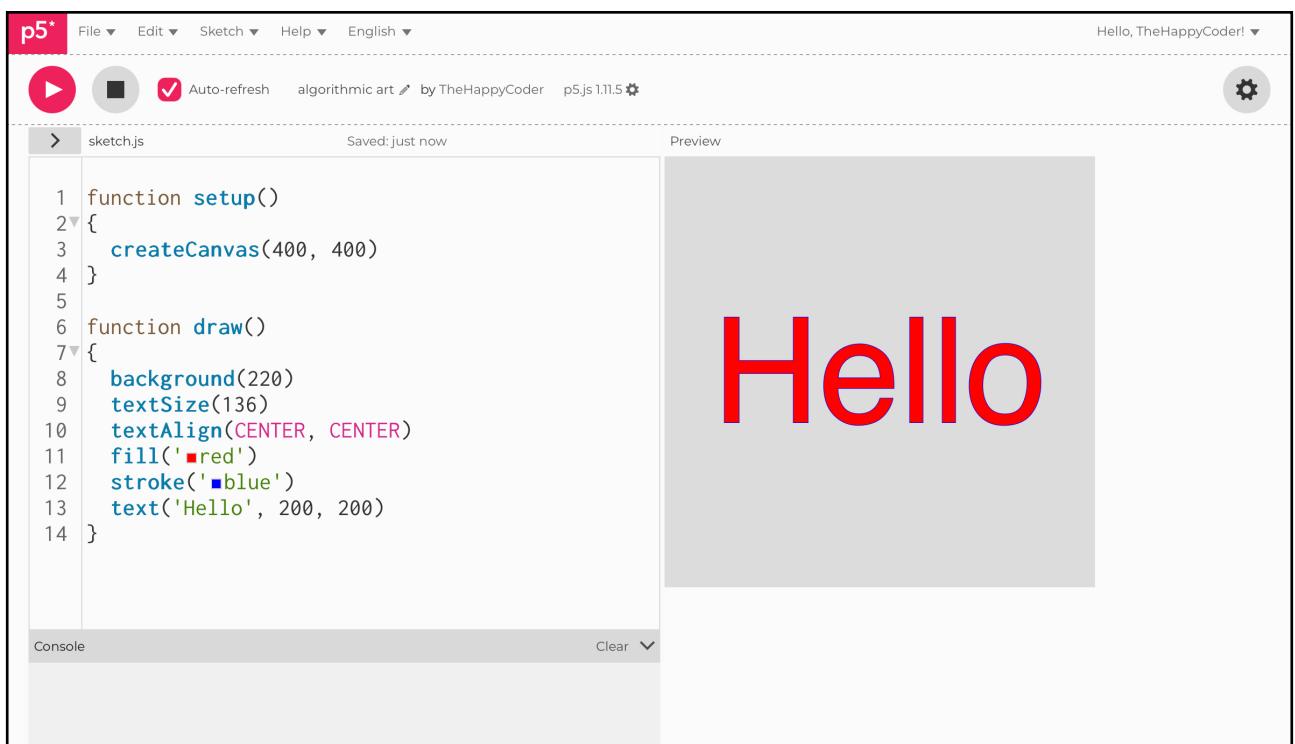
Notes

The blue border around the text is there, it is still only one pixel thick.

Challenge

Alter the **strokeWeight()**.

Figure B3.5



The screenshot shows the p5.js IDE interface. At the top, there's a toolbar with icons for play, stop, refresh, and settings, along with the text "Hello, TheHappyCoder! ▾". The menu bar includes "File", "Edit", "Sketch", "Help", and "English". Below the menu is a status bar showing "sketch.js", "Saved: just now", and "Preview". The main area has two tabs: "sketch.js" (selected) and "Preview". The "sketch.js" tab contains the following code:

```
1 function setup()
2 {
3     createCanvas(400, 400)
4 }
5
6 function draw()
7 {
8     background(220)
9     textSize(136)
10    textAlign(CENTER, CENTER)
11    fill('red')
12    stroke('blue')
13    text('Hello', 200, 200)
14 }
```

The "Preview" tab shows a gray canvas with the word "Hello" centered in large, bold red font.



Sketch B3.6 stroke

! comment out the `fill('red')`

We can do two things: we can increase the `strokeWeight()` and we could have `noFill()`, so we will do both.

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

function draw()
{
    background(220)
    textSize(136)
    textAlign(CENTER, CENTER)
    // fill('red')
    noFill()
    strokeWeight(2)
    stroke('blue')
    text('Hello', 200, 200)
}
```

Notes

Remember that when we comment out `(//)` a line of code, it is ignored.

Challenge

What happens if you don't comment out `fill('red')`?

Figure B3.6

The screenshot shows the p5.js IDE interface. At the top, there's a toolbar with icons for play, stop, and refresh, followed by menu items: File, Edit, Sketch, Help, and English. To the right of the menu is a status bar displaying "Hello, TheHappyCoder! ▾". Below the toolbar, the sketch title is "sketch.js" and it says "Saved: just now". On the left, the code editor contains the following JavaScript code:

```
1 function setup()
2 {
3     createCanvas(400, 400)
4 }
5
6 function draw()
7 {
8     background(220)
9     textSize(136)
10    textAlign(CENTER, CENTER)
11    // fill('red')
12    noFill()
13    strokeWeight(2)
14    stroke('blue')
15    text('Hello', 200, 200)
16 }
```

To the right of the code editor is a preview window titled "Preview" which displays the word "Hello" in large blue outline text centered on a gray background. At the bottom of the interface, there are "Console" and "Clear" buttons.



Sketch B3.7 translate

We have another trick up our sleeve: we can rotate the text. Before we do anything, we need to translate the canvas so that the origin is in the centre of the canvas. We will also need to set the coordinates of the text to **(0, 0)**.

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

function draw()
{
    background(220)
    translate(width/2, height/2)
    textSize(136)
    textAlign(CENTER, CENTER)
    // fill('red')
    noFill()
    strokeWeight(2)
    stroke('blue')
    text('Hello', 0, 0)
}
```

Notes

Everything should be as we had before.

Figure B3.7

The screenshot shows the p5.js IDE interface. At the top, there's a toolbar with icons for play, stop, and refresh, followed by menu items: File, Edit, Sketch, Help, and English. To the right, it says "Hello, TheHappyCoder! ▾". The main area has tabs for "sketch.js" and "Preview". The code editor contains the following JavaScript code:

```
1 function setup()
2 {
3     createCanvas(400, 400)
4 }
5
6 function draw()
7 {
8     background(220)
9     translate(width/2, height/2)
10    textSize(136)
11    textAlign(CENTER, CENTER)
12    // fill('red')
13    noFill()
14    strokeWeight(2)
15    stroke('blue')
16    text('Hello', 0, 0)
17 }
```

The preview window shows a light gray background with the word "Hello" centered in a large, blue, outlined font.



Sketch B3.8 angle of degree

We want an **angle** variable and we will work in degrees, so hence **angleMode()**. We will set the angle to **90°**.

```
let angle = 90

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

function draw()
{
    background(220)
    translate(width/2, height/2)
    rotate(angle)
    textSize(136)
    textAlign(CENTER, CENTER)
    // fill('red')
    noFill()
    strokeWeight(2)
    stroke('blue')
    text('Hello', 0, 0)
}
```

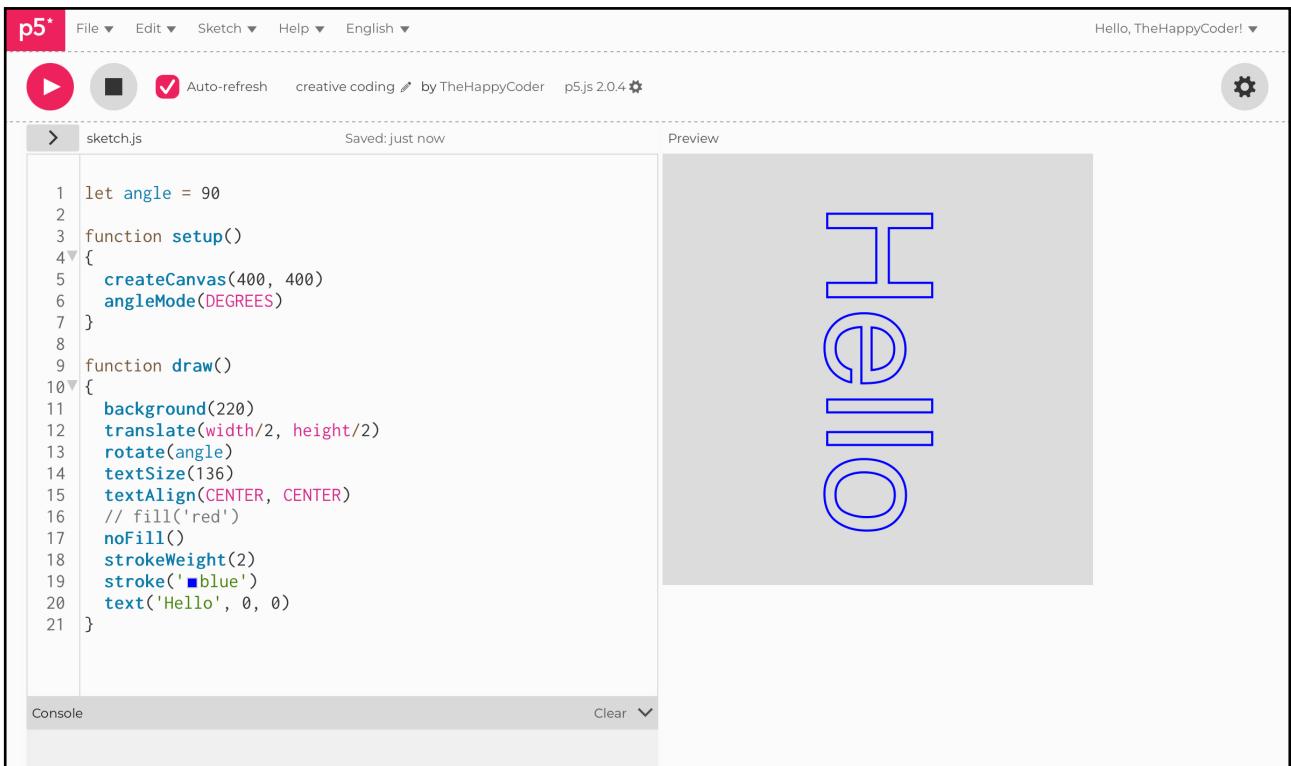
Notes

We have rotated it about the centre of the text.

Challenges

1. Try a different angle.
2. How would you rotate it?

Figure B3.8



The screenshot shows the p5.js code editor interface. At the top, there's a toolbar with icons for play, stop, auto-refresh (which is checked), and user information ('Hello, TheHappyCoder!'). Below the toolbar, the file name 'sketch.js' is displayed along with a 'Saved: just now' message. On the right, there's a preview window showing a gray canvas with blue text and shapes. The code in the editor is as follows:

```
1 let angle = 90
2
3 function setup()
4 {
5   createCanvas(400, 400)
6   angleMode(DEGREES)
7 }
8
9 function draw()
10 {
11   background(220)
12   translate(width/2, height/2)
13   rotate(angle)
14   textSize(136)
15   textAlign(CENTER, CENTER)
16   // fill('red')
17   noFill()
18   strokeWeight(2)
19   stroke('blue')
20   text('Hello', 0, 0)
21 }
```

The preview window shows the word "Hello" rotated 90 degrees counter-clockwise, with each letter consisting of a blue outline. The "H" has a vertical bar on the left and a horizontal bar at the bottom. The "e" has a vertical bar on the left and a horizontal bar at the bottom. The "l" has a vertical bar on the left. The "o" has a vertical bar on the left and a horizontal bar at the bottom.



Sketch B3.9 rotating

We can increment the angle by 1° and rotate it slowly.

```
let angle = 90

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

function draw()
{
    background(220)
    translate(width/2, height/2)
    rotate(angle)
    textSize(136)
    textAlign(CENTER, CENTER)
    // fill('red')
    noFill()
    strokeWeight(2)
    stroke('blue')
    text('Hello', 0, 0)
    angle += 1
}
```

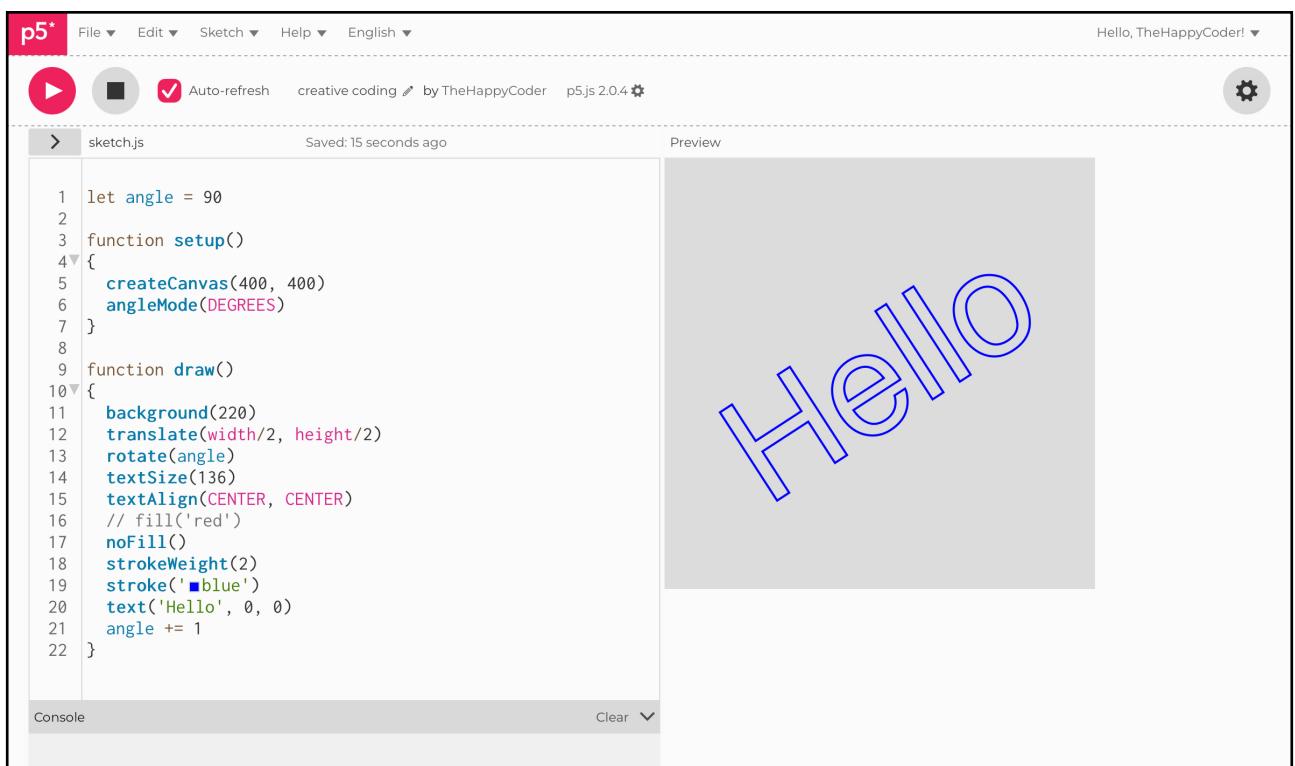
Notes

You should have the text slowly rotating.

 Challenges

1. How would you make it go faster or slower?
2. Change colour as it spins.
3. Change size as it spins.

Figure B3.9



The screenshot shows the p5.js IDE interface. At the top, there's a toolbar with icons for play, stop, auto-refresh (which is checked), and help. The title bar says "sketch.js" and "Saved: 15 seconds ago". On the right, there's a preview window showing a light gray canvas with the word "Hello" written in blue, rotated diagonally from bottom-left to top-right. Below the code editor, there's a "Console" tab and a "Clear" button.

```
let angle = 90
function setup()
{
  createCanvas(400, 400)
  angleMode(DEGREES)
}
function draw()
{
  background(220)
  translate(width/2, height/2)
  rotate(angle)
  textSize(136)
  textAlign(CENTER, CENTER)
  // fill('red')
  noFill()
  strokeWeight(2)
  stroke('blue')
  text('Hello', 0, 0)
  angle += 1
}
```



Sketch B3.10 mouse mover

! Start a new sketch.

We can move text around with the mouse.

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

function draw()
{
    background(220)
    textSize(64)
    text('mouse', mouseX, mouseY)
}
```

Notes

The text should follow the mouse pointer on the canvas.

Challenge

Could you make the text move in the opposite direction to the mouse?

Figure B3.10

The screenshot shows the p5.js IDE interface. At the top, there's a toolbar with icons for play, stop, and refresh, followed by menu items: File, Edit, Sketch, Help, and English. To the right, it says "Hello, TheHappyCoder!" with a dropdown arrow. Below the toolbar, the title bar reads "sketch.js" and "Saved: just now". On the left, the code editor displays the following p5.js code:

```
1 function setup()
2 {
3     createCanvas(400, 400)
4 }
5
6 function draw()
7 {
8     background(220)
9     textSize(64)
10    text('mouse', mouseX, mouseY)
11 }
```

To the right of the code editor is a preview window showing a gray square with the word "mouse" centered in black font. A small circular cursor icon is positioned to the left of the text. At the bottom of the interface, there's a "Console" tab and a "Clear" button.



Sketch B3.11 not a string

We can incorporate values not just strings; here, we will get the **X** coordinate of the mouse on the canvas.

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

function draw()
{
    background(220)
    textSize(64)
    text(mouseX, mouseX, mouseY)
}
```

Notes

We can see the **X** value on the canvas moving with the mouse.

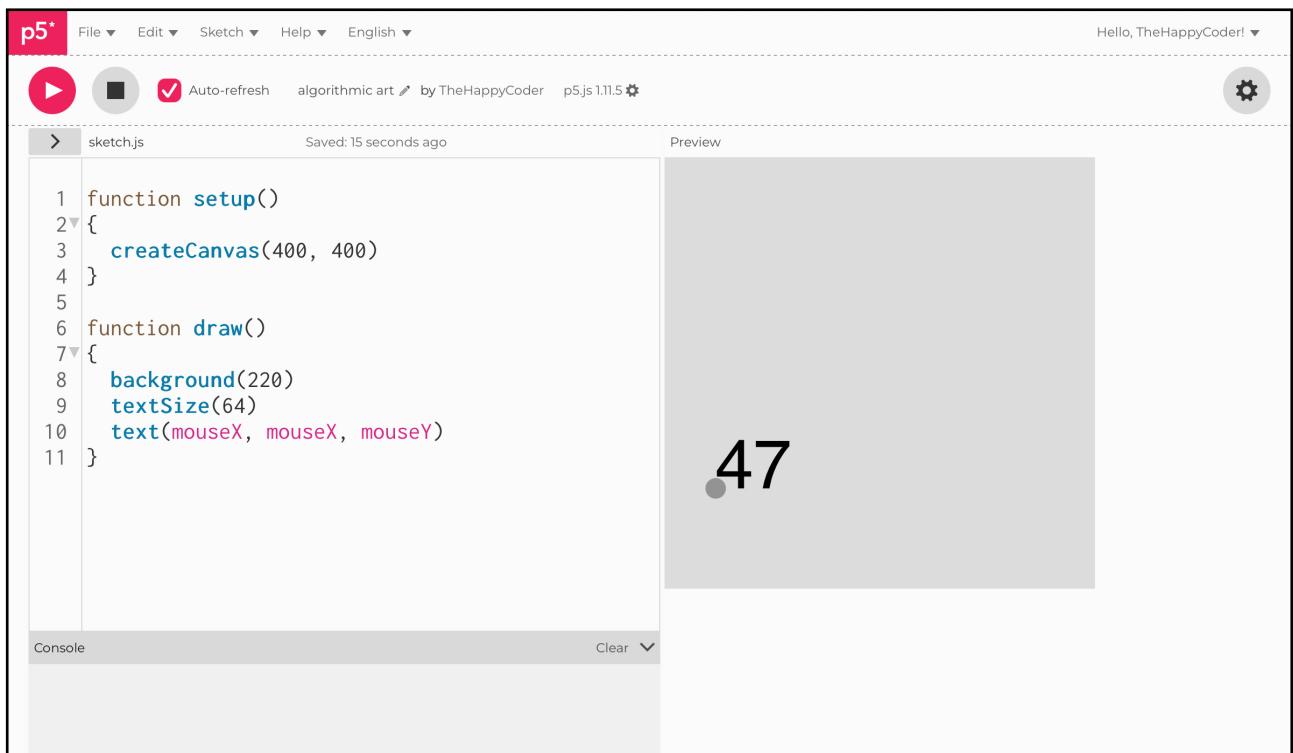
Challenge

Add the **y** value.

Code Explanation

text(mouseX, mouseX, mouseY)	Instead of a string it returns the value of mouseX (first parameter)
------------------------------	--

Figure B3.11



The screenshot shows the p5.js editor interface. At the top, there's a toolbar with icons for play, stop, and refresh, followed by the text "Hello, TheHappyCoder! ▾". Below the toolbar, the file name "sketch.js" is displayed along with a timestamp "Saved: 15 seconds ago". To the right of the file name is a "Preview" button. The main area contains the following JavaScript code:

```
1 function setup()
2 {
3     createCanvas(400, 400)
4 }
5
6 function draw()
7 {
8     background(220)
9     textSize(64)
10    text(mouseX, mouseX, mouseY)
11 }
```

Below the code editor is a "Console" section with a "Clear" dropdown menu.



Sketch B3.12 and mouseY

We can add the mouseY coordinate as well.

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

function draw()
{
    background(220)
    textSize(64)
    text(mouseX, mouseX, mouseY)
    text(mouseY, mouseX, mouseY + 64)
}
```

Notes

We have to separate them, otherwise they will give the numbers on top of each other.

Figure B3.12

The screenshot shows the p5.js IDE interface. At the top, there's a toolbar with icons for play, stop, and refresh, followed by the text "Auto-refresh" with a checked checkbox, "Algorithmic Art" by "TheHappyCoder", and "p5.js 1.11.5". On the right, it says "Hello, TheHappyCoder!" with a dropdown arrow. Below the toolbar, the title bar reads "sketch.js" and "Saved: just now". To the right of the title bar is a "Preview" button. The main area contains the following code:

```
1 function setup()
2 {
3     createCanvas(400, 400)
4 }
5
6 function draw()
7 {
8     background(220)
9     textSize(64)
10    text(mouseX, mouseX, mouseY)
11    text(mouseY, mouseX, mouseY + 64)
12 }
```

The preview window shows a light gray square with the number "161" in large black font at the top-left and "246" in smaller black font below it, with a small gray dot to its left.



Sketch B3.13 static

Instead of following the mouse around, we can have the static values.

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

function draw()
{
    background(220)
    textSize(64)
    text(mouseX, 50, 50)
    text(mouseY, 50, 100)
}
```

Notes

The number changes, but do not move.

Challenge

What happens if you put the **background()** in the **setup()** function?

Figure B3.13

The screenshot shows the p5.js IDE interface. At the top, there's a red header bar with the 'p5*' logo. Below it is a toolbar with icons for play, stop, refresh, and settings. The main area has tabs for 'sketch.js' and 'Preview'. The code editor contains the following JavaScript code:

```
1 function setup()
2 {
3     createCanvas(400, 400)
4 }
5
6 function draw()
7 {
8     background(220)
9     textSize(64)
10    text(mouseX, 50, 50)
11    text(mouseY, 50, 100)
12 }
```

The preview window shows a gray square canvas with the number '60' at (50, 50) and '327' at (50, 100). A small gray dot is visible near the bottom center of the canvas.



Sketch B3.14 adding a string

The values were integers, changing values, we can combine strings (words and letters) and integers.

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

function draw()
{
    background(220)
    textSize(64)
    text('x: ' + mouseX, 50, 50)
    text('y: ' + mouseY, 50, 100)
}
```

Notes

We have both the text and numbers.

Challenge

Put speech marks around `mouseX` and `mouseY`.

Code Explanation

<code>text('x: ' + mouseX, 50, 50)</code>	We can combine text, more text and values together on the same line
---	---

Figure B3.14

The screenshot shows the p5.js code editor interface. At the top, there are menu options: File, Edit, Sketch, Help, and English. On the right side, it says "Hello, TheHappyCoder!" with a dropdown arrow. Below the menu is a toolbar with icons for play, stop, auto-refresh (which is checked), algorithmic art by TheHappyCoder, and p5.js 1.11.5. The main area has tabs for "sketch.js" and "Preview". The code in "sketch.js" is:

```
1 function setup()
2 {
3     createCanvas(400, 400)
4 }
5
6 function draw()
7 {
8     background(220)
9     textSize(64)
10    text('x: ' + mouseX, 50, 50)
11    text('y: ' + mouseY, 50, 100)
12 }
```

The "Preview" window shows a gray square with the text "x: 321" and "y: 342" in large black font. A small gray dot is visible at the bottom center of the canvas. At the bottom of the editor, there is a "Console" tab and a "Clear" button.