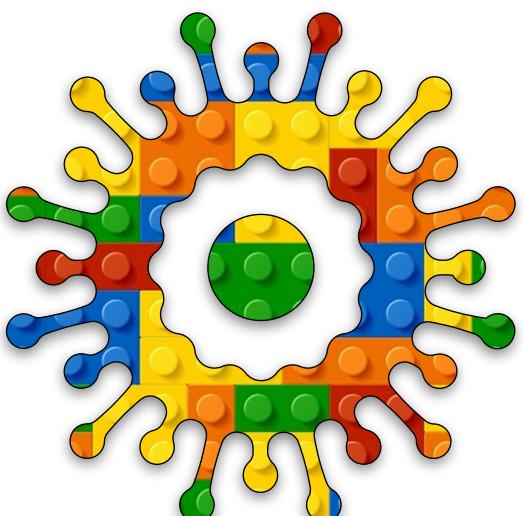


Creative  
Coding  
Module C  
Unit #1

3D shapes





### Module C Unit #1 Primitive 3D shapes

- |              |                            |
|--------------|----------------------------|
| Sketch C1.1  | basic sketch               |
| Sketch C1.2  | adding the WEBGL           |
| Sketch C1.3  | drawing a box              |
| Sketch C1.4  | rotating                   |
| Sketch C1.5  | 45° rotation               |
| Sketch C1.6  | the x and y axis           |
| Sketch C1.7  | incrementing the rotation  |
| Sketch C1.8  | drawing a cuboid           |
| Sketch C1.9  | drawing a plane            |
| Sketch C1.10 | drawing a cylinder         |
| Sketch C1.11 | drawing a sphere           |
| Sketch C1.12 | drawing an ellipsoid       |
| Sketch C1.13 | drawing a torus            |
| Sketch C1.14 | drawing a cone             |
| Sketch C1.15 | adding a bit of colour     |
| Sketch C1.16 | translate                  |
| Sketch C1.17 | translate 'tother way      |
| Sketch C1.18 | translate along the z axis |
| Sketch C1.19 | translate the other way    |
| Sketch C1.20 | more than one shape        |
| Sketch C1.21 | pushing and popping        |



## Introduction to primitive 3D shapes

In p5.js, WEBGL is one of the two available rendering modes, allowing you to create 3D graphics and interactive experiences alongside its default 2D mode.

Key features of WEBGL in p5.js:

- 3D shapes: Draw basic shapes like boxes (cuboids), spheres, cones, cylinders, and more using functions like `box()`, `sphere()`, etc.
- Custom geometry: Create complex models from code or load them from 3D file formats like **OBJ** and **STL**.
- Materials and lighting: Define materials like `basicMaterial()` or `specularMaterial()` and use lights like `ambientLight()` to affect object appearance.
- Camera control: Change the viewpoint using functions like `perspective()`, `ortho()`, and rotate the camera with `rotateX()`, `rotateY()`, `rotateZ()`.
- Textures: Add textures to surfaces for enhanced realism and detail.  
Shaders: For advanced users, write custom shaders to achieve unique visual effects.

We will draw some of the basic shapes and learn about the co-ordinates, translation, and rotation. There are a number of very key differences between the default 2D and the 3D environment.

One of the main changes is how we use the coordinates. All coordinates are based around the centre of the 3D space. There is an **x**, **y**, and **z** component to any coordinates; if only two are specified, then it takes the **x** and **y** and relegates the **z** to zero.

The **x** component is left and right, **y** is top and bottom, and the **z** is front and back.



## Sketch C1.1 standard sketch

Starting with our standard sketch.

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

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



## Sketch C1.2 adding the WEBGL

The first thing to notice is that we have the third argument in the `createCanvas()` function: **WEBGL**.

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

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

### Notes

The first thing you notice is that nothing happens. What you now have is effectively a 3D canvas or space. One where you can draw in the **x**, **y**, and **z** directions.

### Code Explanation

<code>createCanvas(400, 400, WEBGL)</code>	WEGL allows to render an image in 3D
--	--------------------------------------



## Sketch C1.3 drawing a box

What we need is a 3D shape to put into this space. We will start by drawing a **cube**. To draw a **cube**, we use the **box()** function; it will have dimensions of **100** pixels by **100** pixels by **100** pixels. We only need to give the single dimension, as it assumes the other two are the same. We will add the other two when we draw a cuboid later.

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

function draw()
{
    background(220)
    box(100)
}
```

### Notes

One thing you will notice is that I have not specified where to draw it. I have given it no coordinates, yet it has drawn it in the centre of the canvas. Also, it doesn't look very 3D...ish. To the first point, the centre of the canvas is in the centre in all three dimensions, the **x**, **y**, and **z** axes. Secondly, we need to rotate it to see it.

### Challenge

Add the other two dimensions.

### Code Explanation

box(100)	A rectangle with each side of equal length (100)
----------	--

Figure C1.3

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 code editor is a "Preview" window showing a gray canvas with a single white square centered in the middle. The code editor contains the following JavaScript code:

```
1 function setup()
2 {
3     createCanvas(400, 400, WEBGL)
4 }
5
6 function draw()
7 {
8     background(220)
9     box(100)
10 }
```

At the bottom left is a "Console" tab and a "Clear" dropdown menu.



## Sketch C1.4 rotating

We cannot simply rotate it; we have to specify which axis we want to rotate it on, whether it is the **x**, **y**, or **z** axis, and by some angle (measured in **radians**, for now). The functions we use are: **rotateX()**, **rotateY()**, and **rotateZ()**, which I hope are self-explanatory. We will rotate the box along the **x** axis by **2 radians**.

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

function draw()
{
    background(220)
    rotateX(2)
    box(100)
}
```

### Notes

We can now see that it is a cube, at least from one angle.

### Challenge

Try other angles.

### Code Explanation

rotateX(2)	Rotate along the x axis by 2 radians
------------	--------------------------------------

Figure C1.4

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, 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 contains the following JavaScript code:

```
1 function setup()
2 {
3     createCanvas(400, 400, WEBGL)
4 }
5
6 function draw()
7 {
8     background(220)
9     rotateX(2)
10    box(100)
11 }
```

To the right of the code editor is a "Preview" window showing a 3D perspective drawing of a white rectangular box centered on a light gray background. At the bottom of the IDE, there's a "Console" tab and a "Clear" dropdown menu.



## Sketch C1.5 45° rotation

To make life a little easier to understand, I will change the angle mode to **degrees**, which is a bit more intuitive.

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

function draw()
{
    background(220)
    rotateX(45)
    box(100)
}
```

### Notes

Almost the same

### Code Explanation

rotateX(45)	Rotating through 45° along the x axis
-------------	---------------------------------------

Figure C1.5

The screenshot shows the p5.js web editor 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 file name "sketch.js" is shown with a save timestamp: "Saved: about 12 hours ago". On the far right, there's a gear icon for settings.

The main area is divided into two sections: "sketch.js" on the left and "Preview" on the right. The code in "sketch.js" is:

```
1 function setup()
2 {
3   createCanvas(400, 400, WEBGL)
4   angleMode(DEGREES)
5 }
6
7 function draw()
8 {
9   background(220)
10  rotateX(45)
11  box(100)
12 }
```

The "Preview" window shows a 400x400 pixel canvas with a light gray background. A white rectangular box is drawn on the canvas, positioned such that its front face is visible and its back face is partially visible behind it, demonstrating the use of `rotateX` and `box` functions.

At the bottom, there's a "Console" section with a "Clear" button.



## Sketch C1.6 the x and y axis

To show how 3D it is, we are going to rotate along all three axes.

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

function draw()
{
    background(220)
    rotateX(45)
    rotateY(45)
    rotateZ(45)
    box(100)
}
```

### Notes

That is definitely more 3D...ish.

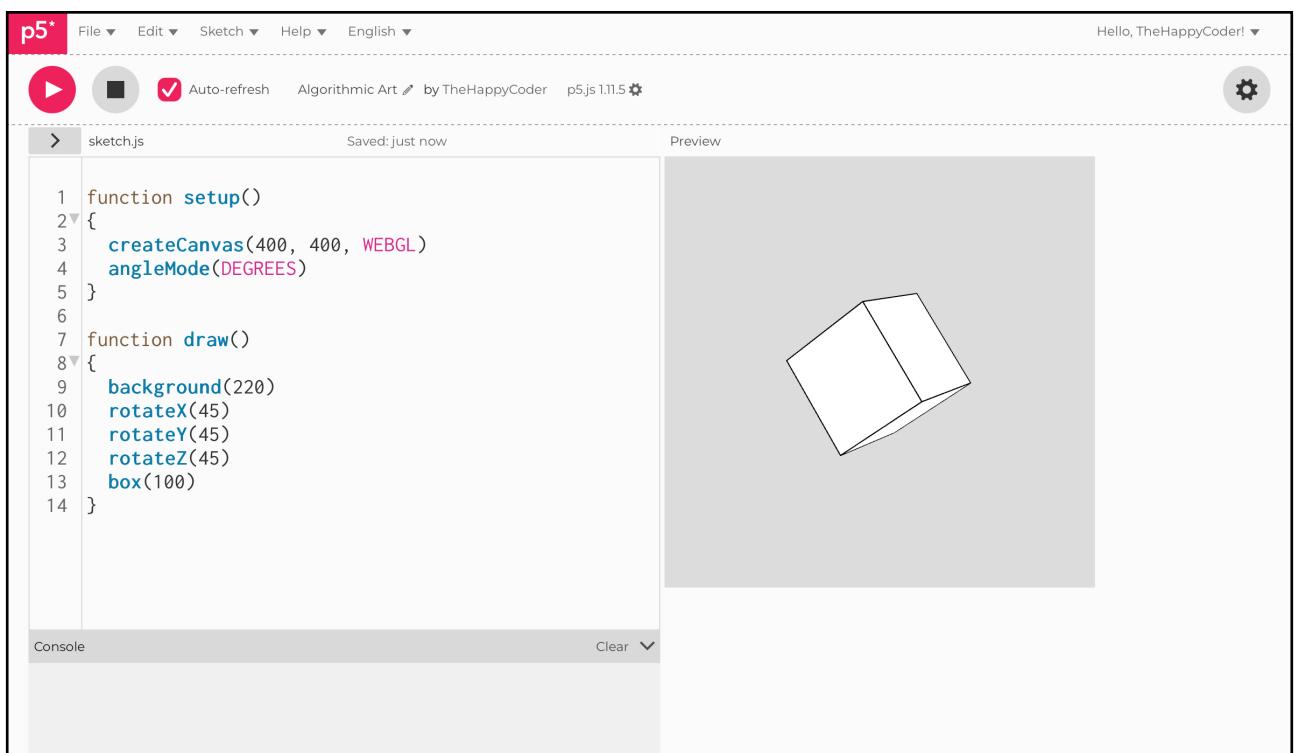
### Challenge

Try different angles.

### Code Explanation

rotateY(45)	Rotating through 45° along the y axis
rotateZ(45)	Rotating through 45° along the z axis

Figure C1.6



A screenshot of the p5.js IDE interface. The top bar includes the p5 logo, file navigation, and user information "Hello, TheHappyCoder! ▾". The main area shows a sketch titled "sketch.js" with the following code:

```
1 function setup()
2 {
3     createCanvas(400, 400, WEBGL)
4     angleMode(DEGREES)
5 }
6
7 function draw()
8 {
9     background(220)
10    rotateX(45)
11    rotateY(45)
12    rotateZ(45)
13    box(100)
14 }
```

The preview window on the right displays a 3D wireframe cube centered in a 400x400 pixel canvas, rotated 45 degrees along all three axes.



## Sketch C1.7 incrementing the rotation

Even better still, we can animate the rotation by introducing a variable (`angle`) and incrementing it.

```
let angle = 0

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

function draw()
{
    background(220)
    rotateX(angle)
    rotateY(angle)
    rotateZ(angle)
    box(100)
    angle++
}
```

### Notes

You should see it rotating in all directions. We have incremented the angle of rotation by  $1^\circ$  on all three axes.

### Challenges

1. Try `-angle` for one of them.
2. Move it faster: `angle += 3`.



## Sketch C1.8 drawing a cuboid

That was just a **cube**; we can add other dimensions to make it a **cuboid**.

```
let angle = 0

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

function draw()
{
    background(220)
    rotateX(angle)
    rotateY(angle)
    rotateZ(angle)
    box(100, 150, 50)
    angle++
}
```

### Notes

Nicely rotating cuboid.

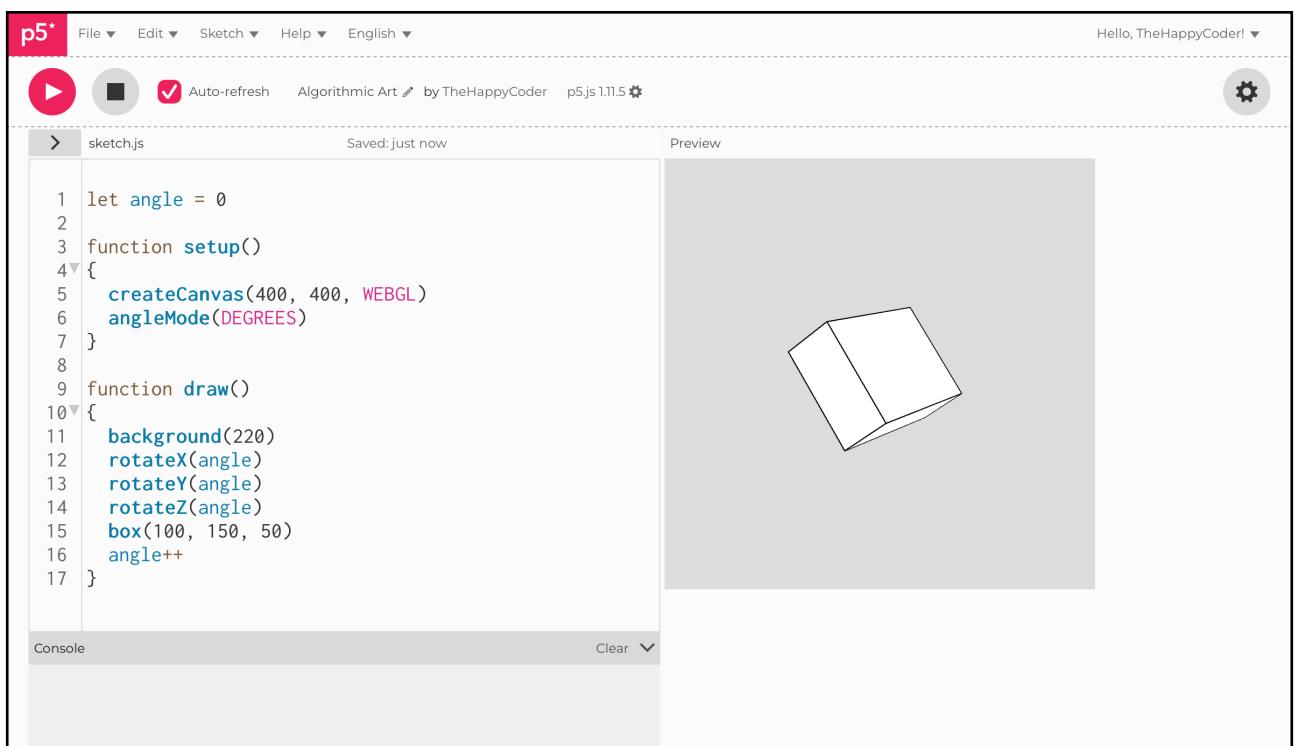
### Challenge

Try other dimensions.

### Code Explanation

box(100, 150, 50)	Draws a cuboid 100 wide, 150 long and 50 deep
-------------------	---

Figure C1.8



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 sketch title is "sketch.js" and it was "Saved: just now". On the left, the code editor contains the following JavaScript code:

```
1 let angle = 0
2
3 function setup()
4{
5  createCanvas(400, 400, WEBGL)
6  angleMode(DEGREES)
7 }
8
9 function draw()
10{
11  background(220)
12  rotateX(angle)
13  rotateY(angle)
14  rotateZ(angle)
15  box(100, 150, 50)
16  angle++
17 }
```

The preview window on the right shows a 3D perspective view of a white cube. The cube is tilted diagonally, demonstrating the effect of the rotation functions. The cube has a bounding box around it.



## Sketch C1.9 drawing a plane

We will draw another shape, a simple **plane**, where it has two dimensions. Do you notice that there is a line going across it? This is because all the shapes that are created are made up of **triangles**.

```
let angle = 0

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

function draw()
{
    background(220)
    rotateX(angle)
    rotateY(angle)
    rotateZ(angle)
    plane(100, 150)
    angle++
}
```

### Notes

A floating plane.

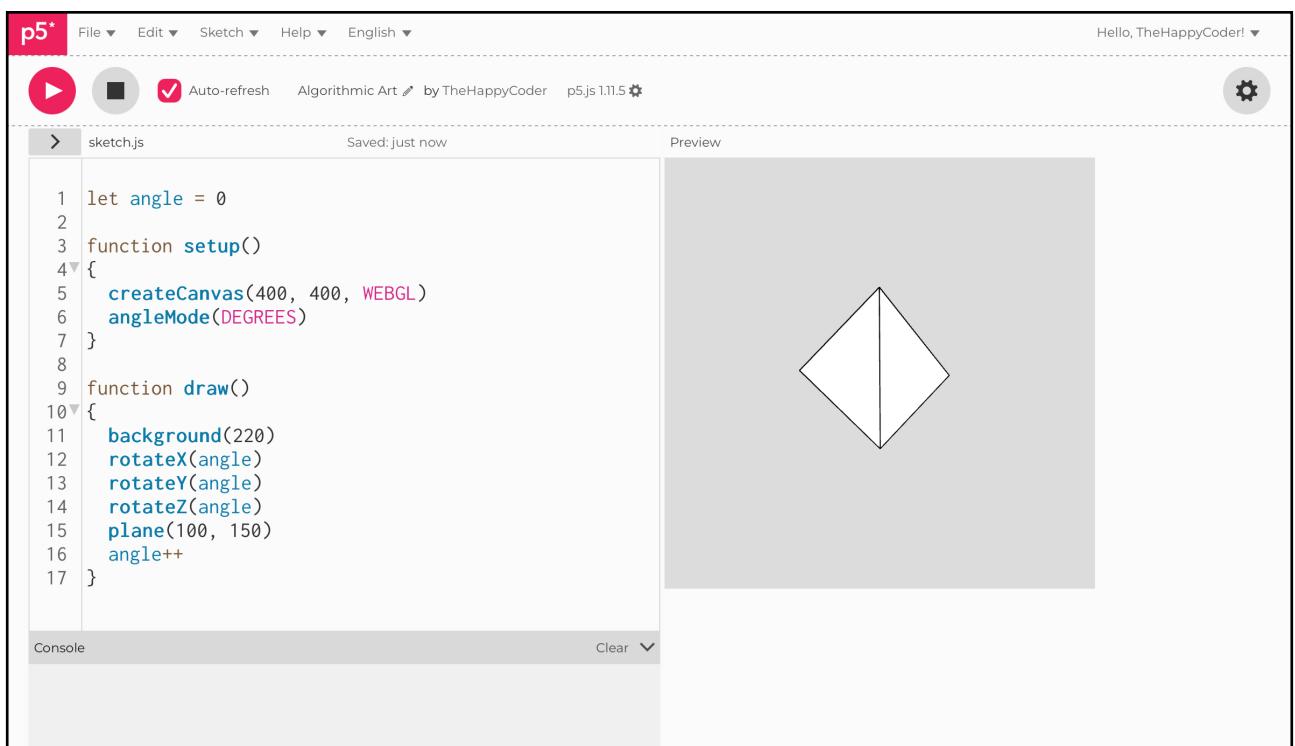
### Challenge

Add **noStroke()**.

### Code Explanation

plane(100, 150)	Creates a flat plane 100 by 150
-----------------	---------------------------------

Figure C1.9



The screenshot shows the p5.js IDE interface. The top bar includes the p5 logo, file navigation, and a user profile. The code editor on the left contains a script named 'sketch.js' with the following code:

```
1 let angle = 0
2
3 function setup()
4{
5  createCanvas(400, 400, WEBGL)
6  angleMode(DEGREES)
7 }
8
9 function draw()
10{
11  background(220)
12  rotateX(angle)
13  rotateY(angle)
14  rotateZ(angle)
15  plane(100, 150)
16  angle++
17 }
```

The preview window on the right displays a 3D diamond shape (a plane) rotated 45 degrees along all three axes. The bottom bar features a 'Console' tab and a 'Clear' button.



## Sketch C1.10 drawing a cylinder

A **cylinder** has two dimensions, one for the radius and the other for the length.

```
let angle = 0

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

function draw()
{
    background(220)
    rotateX(angle)
    rotateY(angle)
    rotateZ(angle)
    cylinder(100, 150)
    angle++
}
```

### Notes

The first is the diameter and the second is the length. If you use **noStroke()** you lose a lot of definition; we will use lights to address that issue (in another unit).

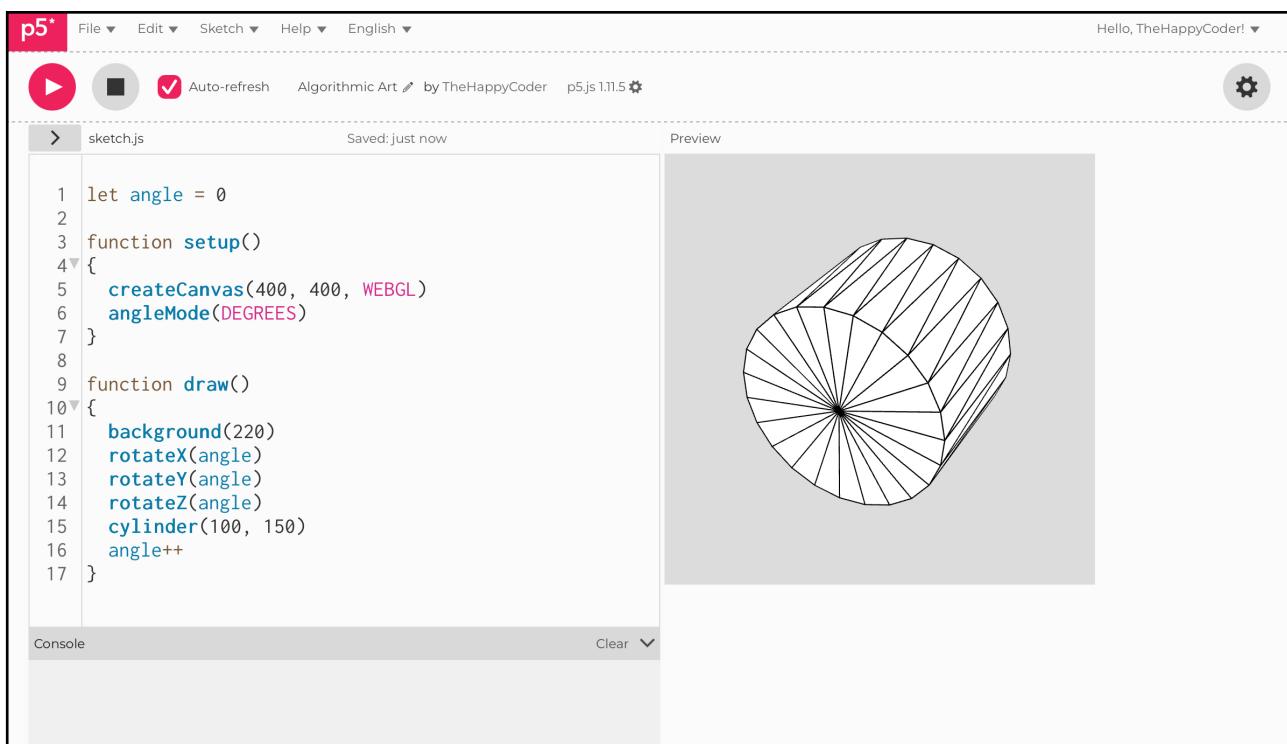
### Challenge

Alter the dimensions.

### Code Explanation

<code>cylinder(100, 150)</code>	A cylinder with a radius 100 and length 150
---------------------------------	---

Figure C1.10



The screenshot shows the p5.js IDE interface. At the top, there's a toolbar with icons for play, stop, and refresh, followed by the text "Auto-refresh" with a checked checkbox, "Algorithmic Art" by TheHappyCoder, and the version "p5.js 1.11.5". On the right, it says "Hello, TheHappyCoder!" with a dropdown arrow. Below the toolbar, the file name "sketch.js" is shown along with "Saved: just now" and a "Preview" button.

The code editor contains the following JavaScript code:

```
let angle = 0

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

}

function draw()
{
  background(220)
  rotateX(angle)
  rotateY(angle)
  rotateZ(angle)
  cylinder(100, 150)
  angle++
}
```

The preview window on the right displays a 3D cylinder rotating in perspective. The cylinder is divided into several vertical slices, and the top face is also subdivided into smaller triangles, creating a faceted appearance. The cylinder rotates around its vertical axis.



## Sketch C1.11 drawing a sphere

Here is a **sphere** with a single dimension, the radius.

```
let angle = 0

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

function draw()
{
    background(220)
    rotateX(angle)
    rotateY(angle)
    rotateZ(angle)
    sphere(100)
    angle++
}
```

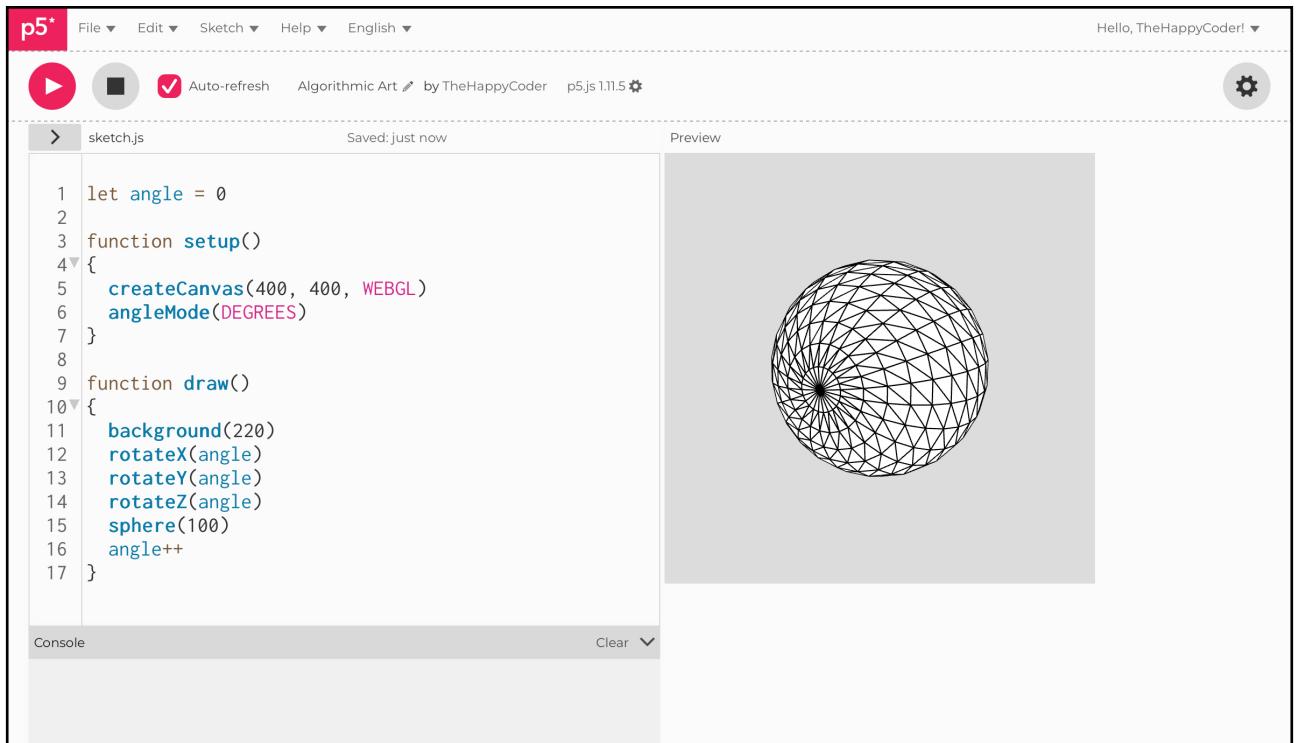
### Notes

We will explore how to make lots of shapes in different positions at a later date.

### Code Explanation

sphere(100)	Sphere with a radius of 100
-------------	-----------------------------

Figure C1.11



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 sketch title is "sketch.js" and it was "Saved: just now". On the left, the code editor contains the following JavaScript code:

```
1 let angle = 0
2
3 function setup()
4{
5  createCanvas(400, 400, WEBGL)
6  angleMode(DEGREES)
7 }
8
9 function draw()
10{
11  background(220)
12  rotateX(angle)
13  rotateY(angle)
14  rotateZ(angle)
15  sphere(100)
16  angle++
17 }
```

The preview window on the right displays a wireframe sphere centered in a 400x400 pixel canvas. The sphere is composed of many small triangles forming a spherical mesh. The background is a light gray.



## Sketch C1.12 drawing an ellipsoid

An **ellipsoid** is a sphere but with two dimension. A radius in one direction and a perpendicular radius.

```
let angle = 0

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

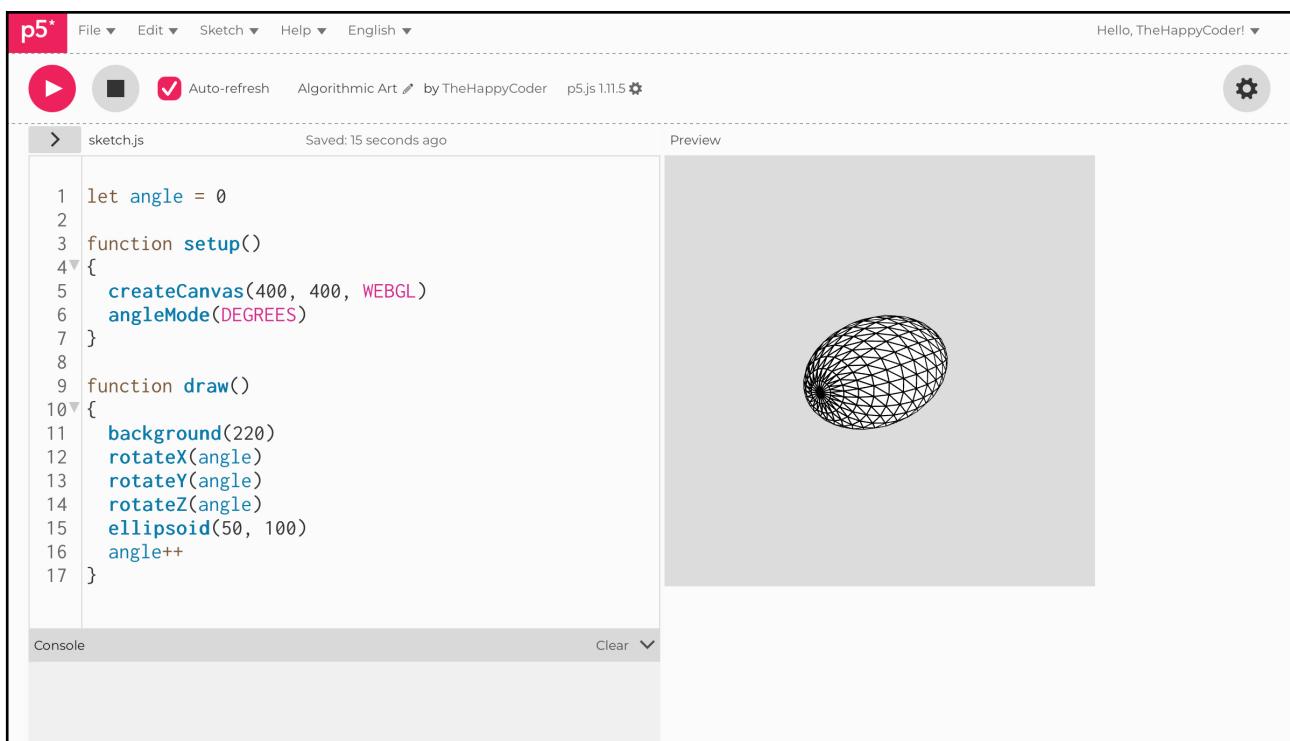
function draw()
{
    background(220)
    rotateX(angle)
    rotateY(angle)
    rotateZ(angle)
    ellipsoid(50, 100)
    angle++
}
```

### 🔧 Code Explanation

`ellipsoid(50, 100)`

Drawing an ellipsoid 50 wide, 100 long

Figure C1.12



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 other options like Sketch, Help, and English. To the right of the toolbar, it says "Hello, TheHappyCoder! ▾". Below the toolbar, the file name is "sketch.js" and it was saved 15 seconds ago. On the right side, there's a preview window showing a wireframe ellipsoid. The code in the editor is:

```
1 let angle = 0
2
3 function setup()
4{
5  createCanvas(400, 400, WEBGL)
6  angleMode(DEGREES)
7 }
8
9 function draw()
10{
11  background(220)
12  rotateX(angle)
13  rotateY(angle)
14  rotateZ(angle)
15  ellipsoid(50, 100)
16  angle++
17 }
```

At the bottom left is a "Console" button and at the bottom right is a "Clear" dropdown menu.



## Sketch C1.13 drawing a torus

Now for a **torus** (doughnut/donut), which has two dimensions: the torus radius and tube radius (thickness of the doughnut).

```
let angle = 0

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

function draw()
{
    background(220)
    rotateX(angle)
    rotateY(angle)
    rotateZ(angle)
    torus(100, 50)
    angle++
}
```

### Notes

The radius of the torus (taken at the centre of the tube) is the first argument, and the radius of the actual tube itself is the second.

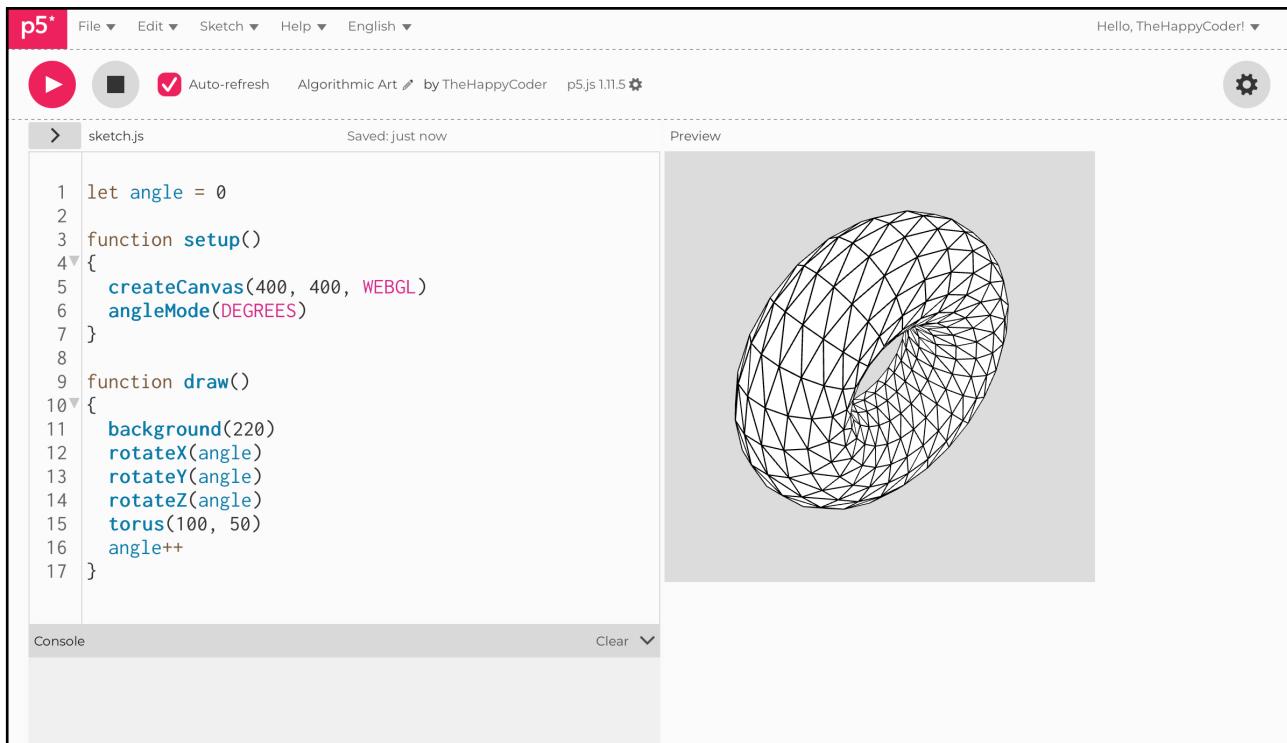
### Challenge

Play with the values to get a feel for the shape.

### Code Explanation

torus(100, 50)	A torus with a radius of 100 and a tube dimension of 50
----------------	---

Figure C1.13



The screenshot shows the p5.js IDE interface. At the top, there's a menu bar with 'File ▾', 'Edit ▾', 'Sketch ▾', 'Help ▾', and 'English ▾'. To the right of the menu is a greeting 'Hello, TheHappyCoder! ▾' and a gear icon. Below the menu, there are three buttons: a play button, a square button, and an auto-refresh button with a checkmark. The title bar says 'sketch.js' and 'Saved: just now'. On the right side, there's a 'Preview' window showing a wireframe torus (doughnut shape) with a complex triangular mesh. The main workspace contains the following p5.js code:

```
1 let angle = 0
2
3 function setup()
4 {
5   createCanvas(400, 400, WEBGL)
6   angleMode(DEGREES)
7 }
8
9 function draw()
10 {
11   background(220)
12   rotateX(angle)
13   rotateY(angle)
14   rotateZ(angle)
15   torus(100, 50)
16   angle++
17 }
```

At the bottom left is a 'Console' tab and at the bottom right is a 'Clear ▾' button.



## Sketch C1.14 drawing a cone

A **cone** which has two dimensions, also, the first is the radius of the base and the second is the length (or height) of the cone.

```
let angle = 0

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

function draw()
{
    background(220)
    rotateX(angle)
    rotateY(angle)
    rotateZ(angle)
    cone(100, 200)
    angle++
}
```

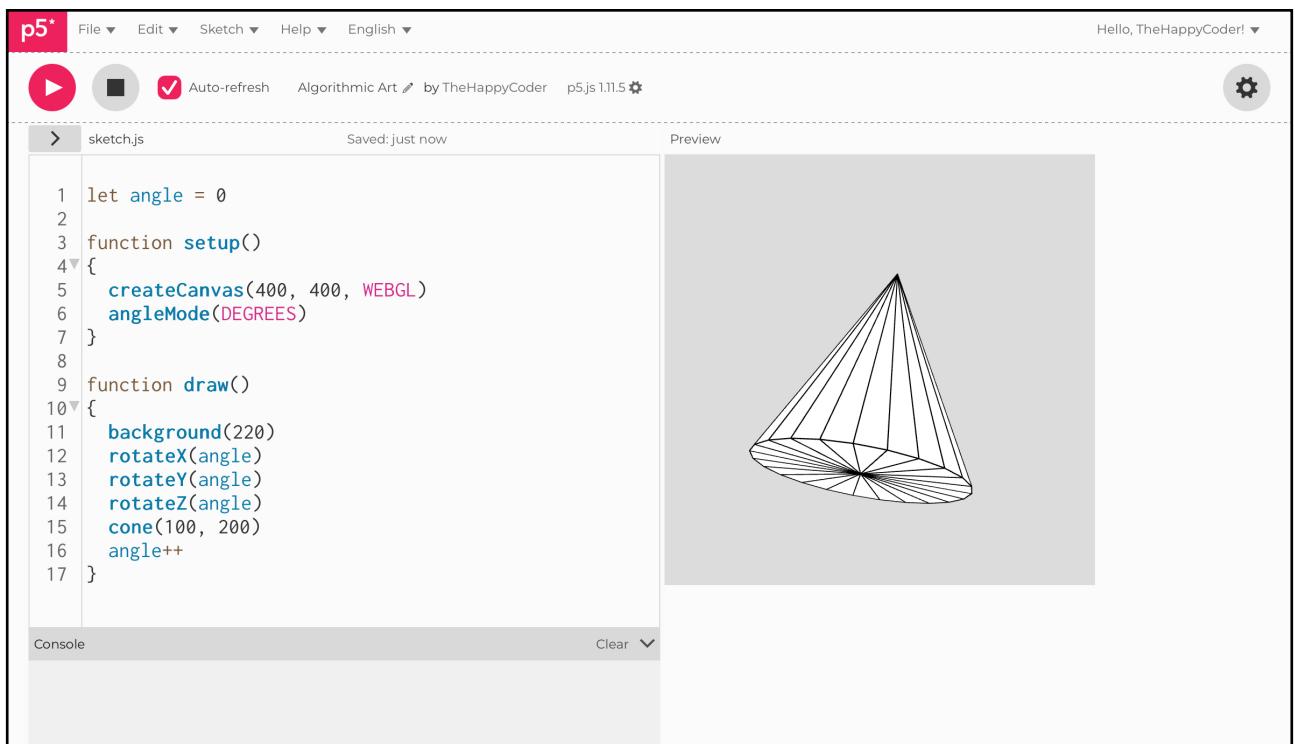
### Challenges

1. Have more than one shape at the same time.
2. Use **noFill()**.

### Code Explanation

cone(100, 200)	Draws a cone with a radius 100 and length 200
----------------	---

Figure C.14



The screenshot shows the p5.js web-based IDE interface. At the top, there's a toolbar with icons for play/pause, stop, refresh, and settings, along with the text "Hello, TheHappyCoder! ▾". The main area has tabs for "sketch.js" and "Preview". The code editor contains the following JavaScript code:

```
let angle = 0

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

function draw()
{
    background(220)
    rotateX(angle)
    rotateY(angle)
    rotateZ(angle)
    cone(100, 200)
    angle++
}
```

The preview window on the right displays a 3D cone rotating in perspective. The cone is rendered with fine lines, creating a faceted appearance. It rotates around its vertical axis, which is positioned towards the bottom left of the canvas.



## Sketch C1.15 adding a bit of colour

! remove cone()

Let's go back to our simple cube. We can add colour with **fill()** and give the **background()** some colour for good measure.

```
let angle = 0

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

function draw()
{
    background('darkred')
    fill('yellow')
    rotateX(angle)
    rotateY(angle)
    rotateZ(angle)
    box(100)
    angle++
}
```

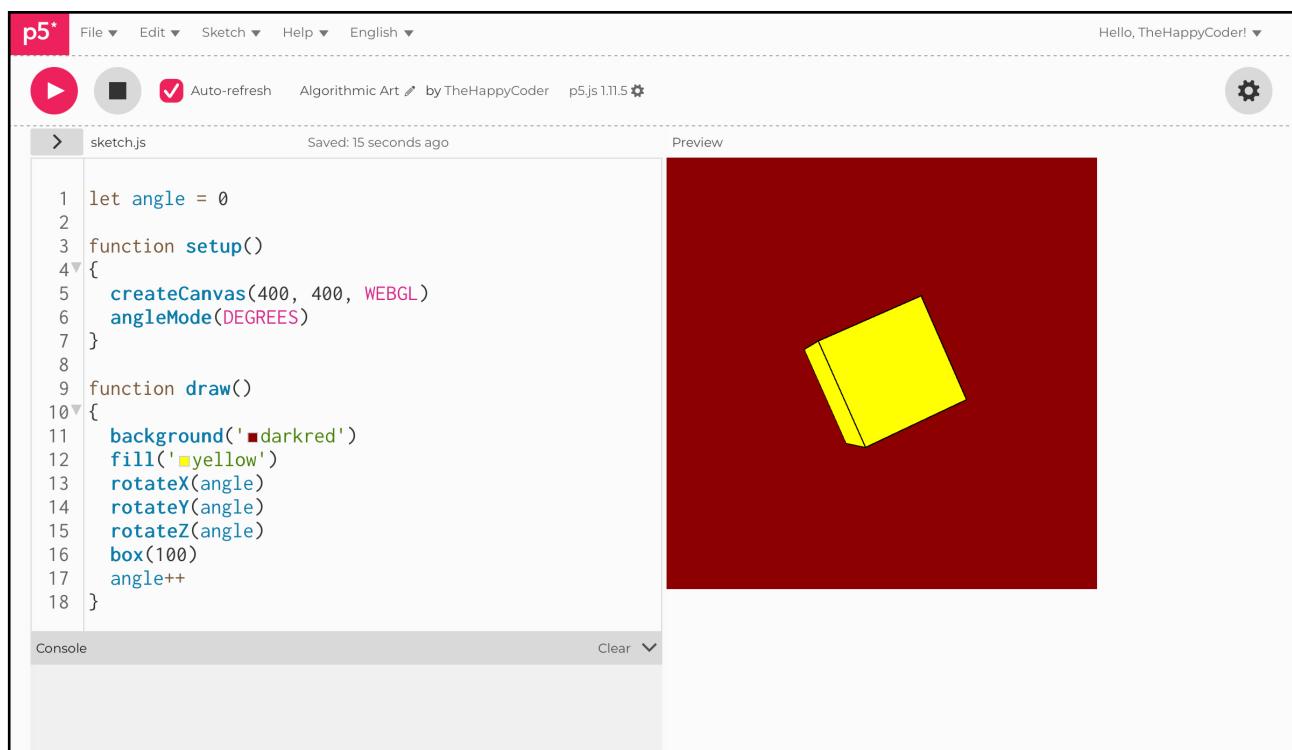
### Notes

This gives us a nice dark red background and a yellow cube. We will look at materials later.

### Challenge

Explore other colours.

Figure C1.15



The screenshot shows the p5.js web-based IDE interface. At the top, there's a toolbar with icons for play/pause, stop, refresh, and settings, along with the text "Hello, TheHappyCoder! ▾". Below the toolbar, the file name "sketch.js" and the message "Saved: 15 seconds ago" are displayed. The main area is divided into two sections: "sketch.js" on the left containing the code, and "Preview" on the right showing the resulting 3D visualization.

**sketch.js**

```
1 let angle = 0
2
3 function setup()
4{
5  createCanvas(400, 400, WEBGL)
6  angleMode(DEGREES)
7 }
8
9 function draw()
10{
11  background('darkred')
12  fill('yellow')
13  rotateX(angle)
14  rotateY(angle)
15  rotateZ(angle)
16  box(100)
17  angle++
18 }
```

**Preview**

A 3D perspective view of a single yellow cube centered in a dark red square frame. The cube is tilted diagonally, demonstrating the effect of the combined 3D transformations in the code.



## Sketch C1.16 translate

To move a shape in 3D (**WEBGL**), we translate the origin of the space. It takes a bit of getting used to as we are translating relative to the centre of the 3D space rather than from the top left as in the 2D canvas.

```
let angle = 0

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

function draw()
{
    background('darkred')
    fill('yellow')
    translate(100, 100) // This line is highlighted in light blue
    rotateX(angle)
    rotateY(angle)
    rotateZ(angle)
    box(100)
    angle++
}
```

### Notes

This moves it down and to the right; notice that it still rotates about the new origin.

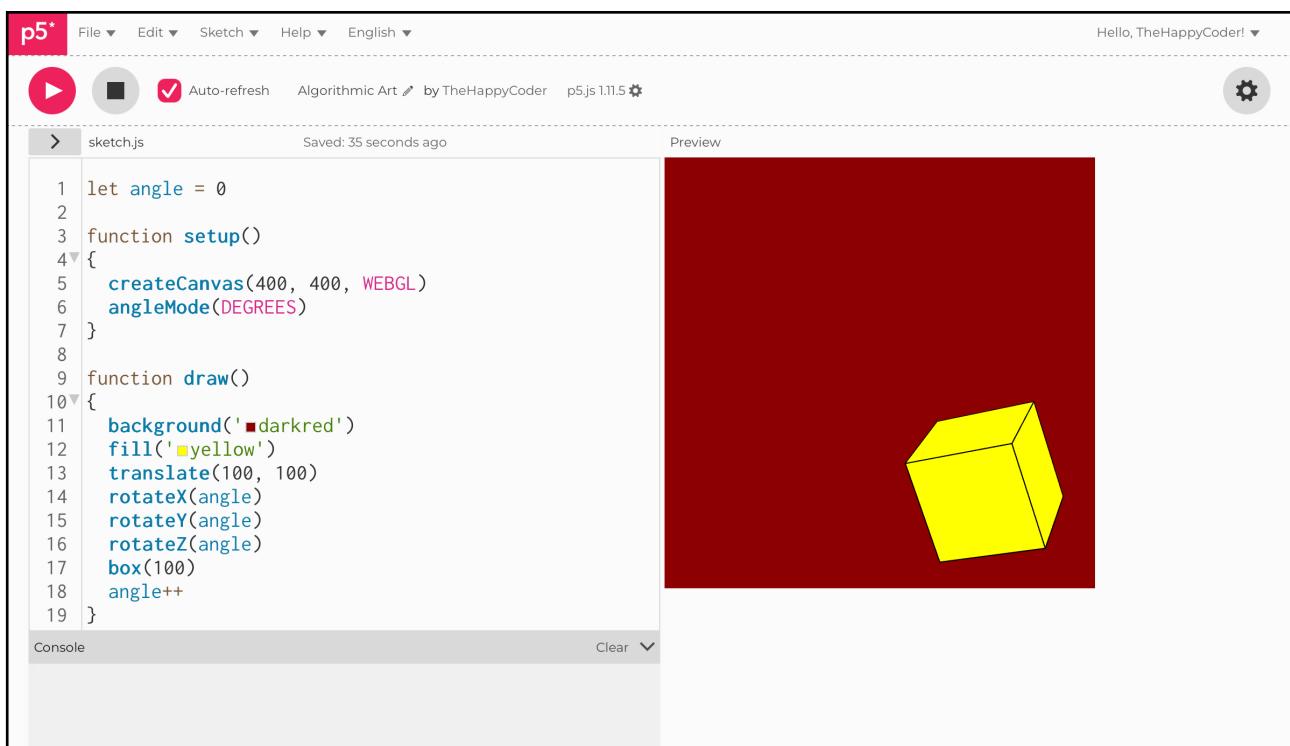
### Challenges

1. Try other translations.
2. Do you know how to translate it to the left?

## Code Explanation

translate(100, 100)	Translating relative to the original origin
---------------------	---

Figure C1.16



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: 35 seconds ago". On the right, it says "Hello, TheHappyCoder! ▾" and has a gear icon. The main area is divided into two sections: "sketch.js" on the left containing the code, and "Preview" on the right showing the output.

```
let angle = 0
function setup()
{
  createCanvas(400, 400, WEBGL)
  angleMode(DEGREES)
}
function draw()
{
  background('darkred')
  fill('yellow')
  translate(100, 100)
  rotateX(angle)
  rotateY(angle)
  rotateZ(angle)
  box(100)
  angle++
}
```

The preview window shows a 3D yellow cube centered on a dark red background. The cube is oriented diagonally, with its front face facing towards the bottom-left. The code uses p5.js's built-in WebGL renderer to create the 3D effect.



## Sketch C1.17 translate 'tother way

Translating it to the top left-hand corner

```
let angle = 0

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

function draw()
{
    background('darkred')
    fill('yellow')
    translate(-100, -100)
    rotateX(angle)
    rotateY(angle)
    rotateZ(angle)
    box(100)
    angle++
}
```

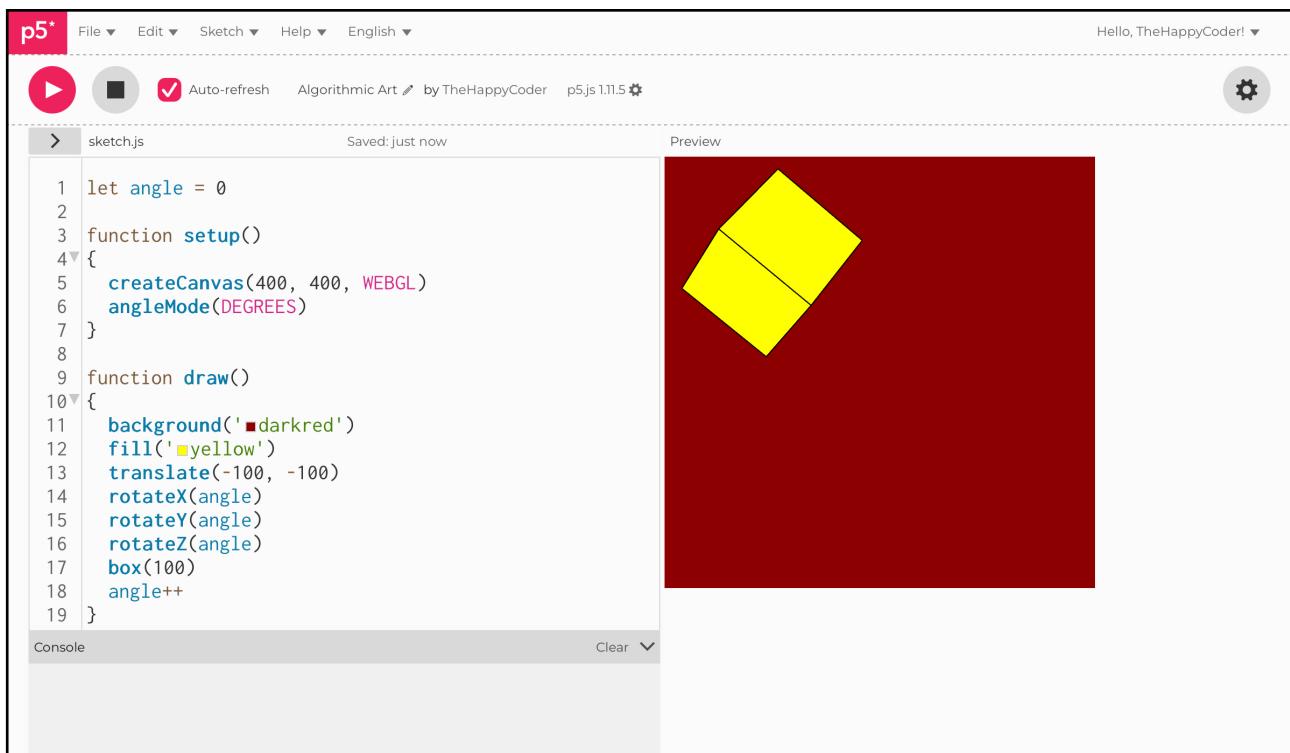
### Notes

We need to translate it negatively relative to the old origin.

### Code Explanation

translate(-100, -100)	Translating relative to the original origin
-----------------------	---

Figure C1.17



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 sketch name "sketch.js" is shown along with a "Saved: just now" message. On the left is the code editor with the following P5.js code:

```
1 let angle = 0
2
3 function setup()
4 {
5   createCanvas(400, 400, WEBGL)
6   angleMode(DEGREES)
7 }
8
9 function draw()
10 {
11   background('darkred')
12   fill('yellow')
13   translate(-100, -100)
14   rotateX(angle)
15   rotateY(angle)
16   rotateZ(angle)
17   box(100)
18   angle++
19 }
```

To the right of the code editor is a preview window titled "Preview" showing a 400x400 pixel canvas. The canvas has a dark red background. A single yellow cube is centered on the canvas, tilted diagonally. It appears to be rotating, as indicated by the perspective and the angle variable in the code. The bottom of the preview window has a "Clear" button with a dropdown arrow.



## Sketch C1.18 translate along the z axis

But what about the **z** axis? We will now translate along the **z** axis only; we still need all three arguments for translate.

```
let angle = 0

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

function draw()
{
    background('darkred')
    fill('yellow')
    translate(0, 0, 500) // This line is highlighted in blue
    rotateX(angle)
    rotateY(angle)
    rotateZ(angle)
    box(100)
    angle++
}
```

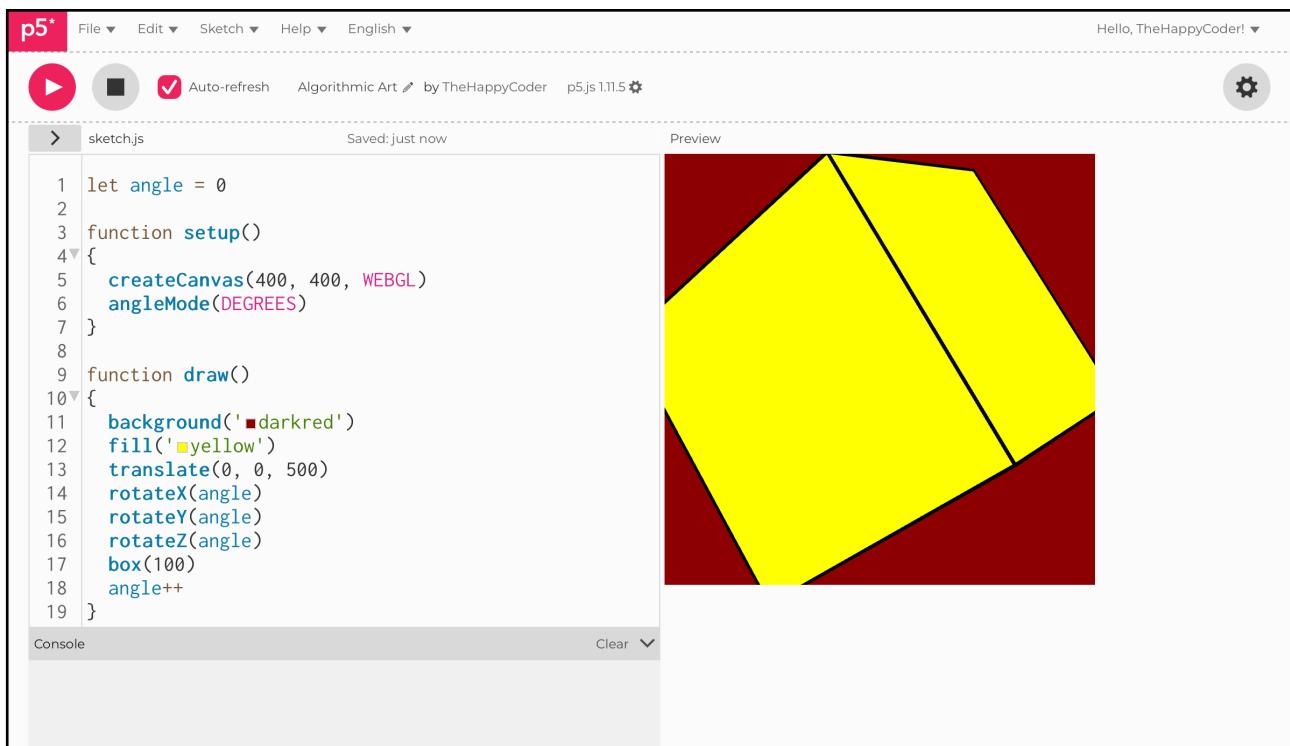
### Notes

This has moved it a lot closer; a positive value moves it towards you.

### Code Explanation

translate(0, 0, 500)	Translated only in the z direction, positive value brings it forwards
----------------------	---

Figure C1.18



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 settings. The title bar says "sketch.js" and "Saved: just now". On the right, it says "Hello, TheHappyCoder! ▾". Below the toolbar, there's a code editor with the following JavaScript code:

```
1 let angle = 0
2
3 function setup()
4 {
5   createCanvas(400, 400, WEBGL)
6   angleMode(DEGREES)
7 }
8
9 function draw()
10 {
11   background('darkred')
12   fill('yellow')
13   translate(0, 0, 500)
14   rotateX(angle)
15   rotateY(angle)
16   rotateZ(angle)
17   box(100)
18   angle++
19 }
```

To the right of the code editor is a "Preview" window showing a 3D perspective view of a yellow cube. The cube is positioned in the center of a dark red rectangular plane. The cube is oriented diagonally, with its front face facing towards the bottom-left. A single black line segment connects the top-right vertex of the front face to the top-left vertex of the back face. The rest of the cube and the plane are rendered in a solid yellow color.



## Sketch C1.19 translate the other way

Giving it a negative value moves it further away; you are translating the space, not the shape, remember. So if you add other shapes, they too will be affected by the same amount.

```
let angle = 0

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

function draw()
{
    background('darkred')
    fill('yellow')
    translate(0, 0, -500) // This line is highlighted in light blue
    rotateX(angle)
    rotateY(angle)
    rotateZ(angle)
    box(100)
    angle++
}
```

### Notes

You can see how you can position the shape relative to the origin in all three planes (axes).

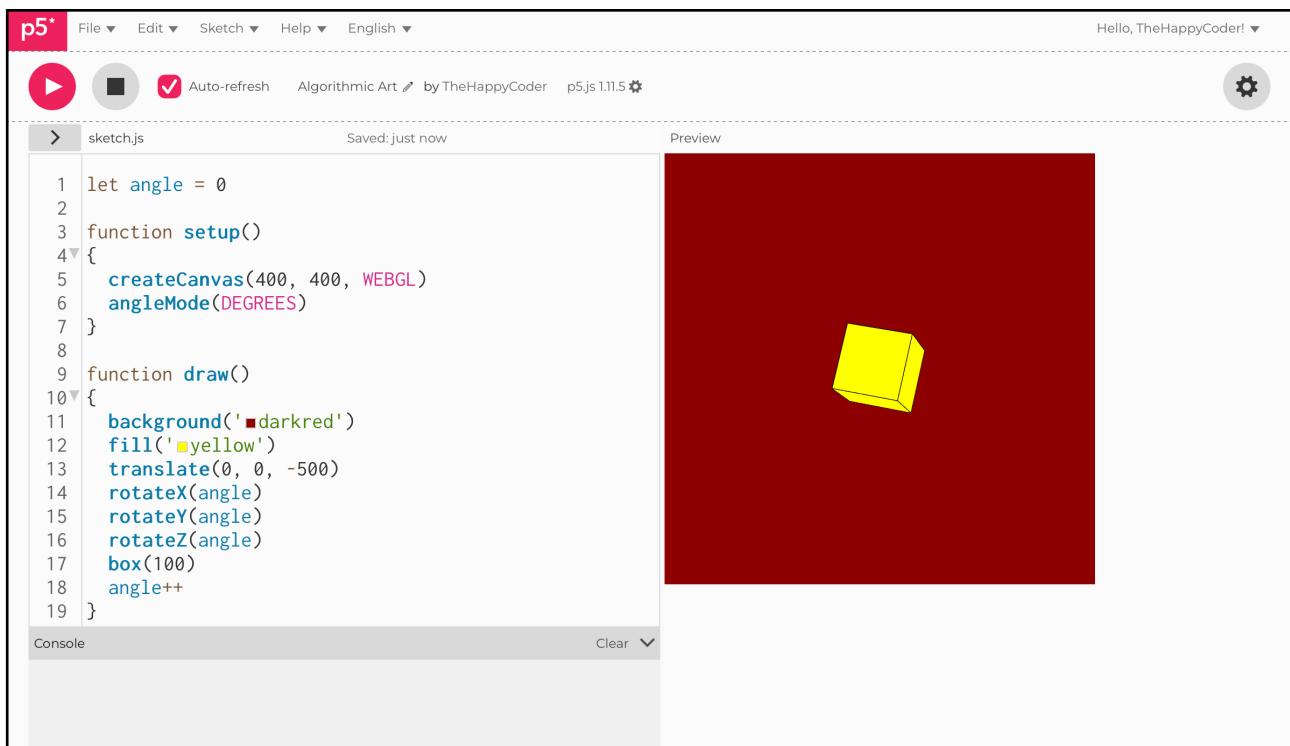
### Challenge

Play with all three values.

## Code Explanation

translate(0, 0, -500)	Moves it further away by 500
-----------------------	------------------------------

Figure C1.19



The screenshot shows the p5.js web-based IDE interface. At the top, there's a toolbar with icons for play/pause, stop, refresh, and settings, along with the text "Hello, TheHappyCoder! ▾". The main area has tabs for "sketch.js" and "Preview". The code editor contains the following JavaScript code:

```
1 let angle = 0
2
3 function setup()
4 {
5   createCanvas(400, 400, WEBGL)
6   angleMode(DEGREES)
7 }
8
9 function draw()
10 {
11   background('darkred')
12   fill('yellow')
13   translate(0, 0, -500)
14   rotateX(angle)
15   rotateY(angle)
16   rotateZ(angle)
17   box(100)
18   angle++
19 }
```

The preview window on the right shows a 3D perspective view of a single yellow cube centered in a dark red square frame. The cube is oriented diagonally, with its front face facing towards the viewer.



## Sketch C1.20 more than one shape

But what happens if you have more than one shape occupying the same coordinates? Let's find out. We will add a cone and a torus to the box.

**! Remove `translate()`**

```
let angle = 0

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

function draw()
{
    background('darkred')
    fill('yellow')
    // translate(0, 0, -500)
    rotateX(angle)
    rotateY(angle)
    rotateZ(angle)
    box(100)
    torus(100, 25)
    cone(100, 200)
    angle++
}
```

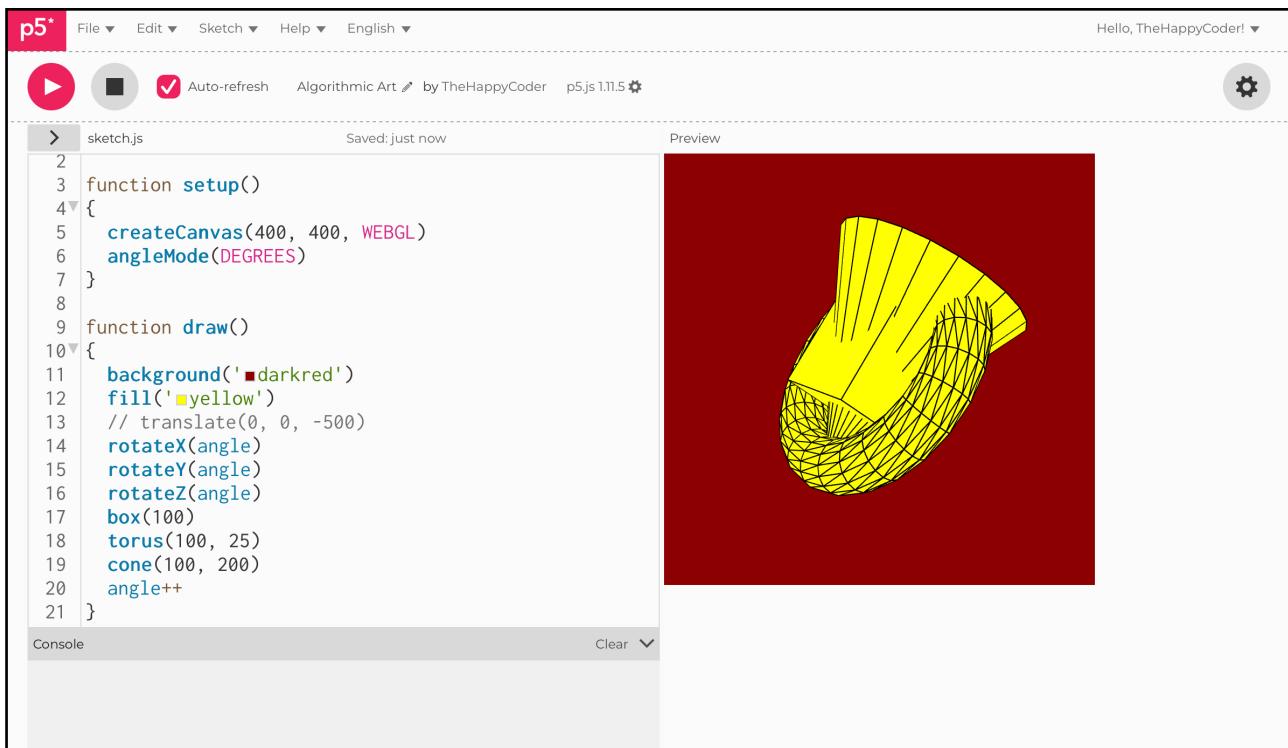
### Notes

All three are meshed together.

### Challenge

Try `noFill()` and `stroke('white')`.

Figure C1.20



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! ▾". Below the menu is a toolbar with a play button, a square, a checkmark, and other icons. The main area has tabs for "sketch.js" and "Preview". The code in "sketch.js" is as follows:

```
2
3 function setup()
4 {
5   createCanvas(400, 400, WEBGL)
6   angleMode(DEGREES)
7 }
8
9 function draw()
10 {
11   background('darkred')
12   fill('yellow')
13   // translate(0, 0, -500)
14   rotateX(angle)
15   rotateY(angle)
16   rotateZ(angle)
17   box(100)
18   torus(100, 25)
19   cone(100, 200)
20   angle++
21 }
```

The "Preview" window shows a 3D wireframe model of a torus-like shape, colored yellow, against a dark red background. The model has a complex internal structure with many intersecting lines.



## Sketch C1.21 pushing and popping

Using `push()` and `pop()` we can rotate and translate individual shapes (or groups of shapes).

```
let angle = 0

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

function draw()
{
    background('darkred')
    fill('yellow')
    // translate(0, 0, -500)
    rotateX(angle)
    rotateY(angle)
    rotateZ(angle)
    box(100)

    push()
    translate(0, 0, 100)
    rotateX(angle * 2)
    torus(100, 25)

    pop()

    cone(100, 200)
    angle++
}
```



## Notes

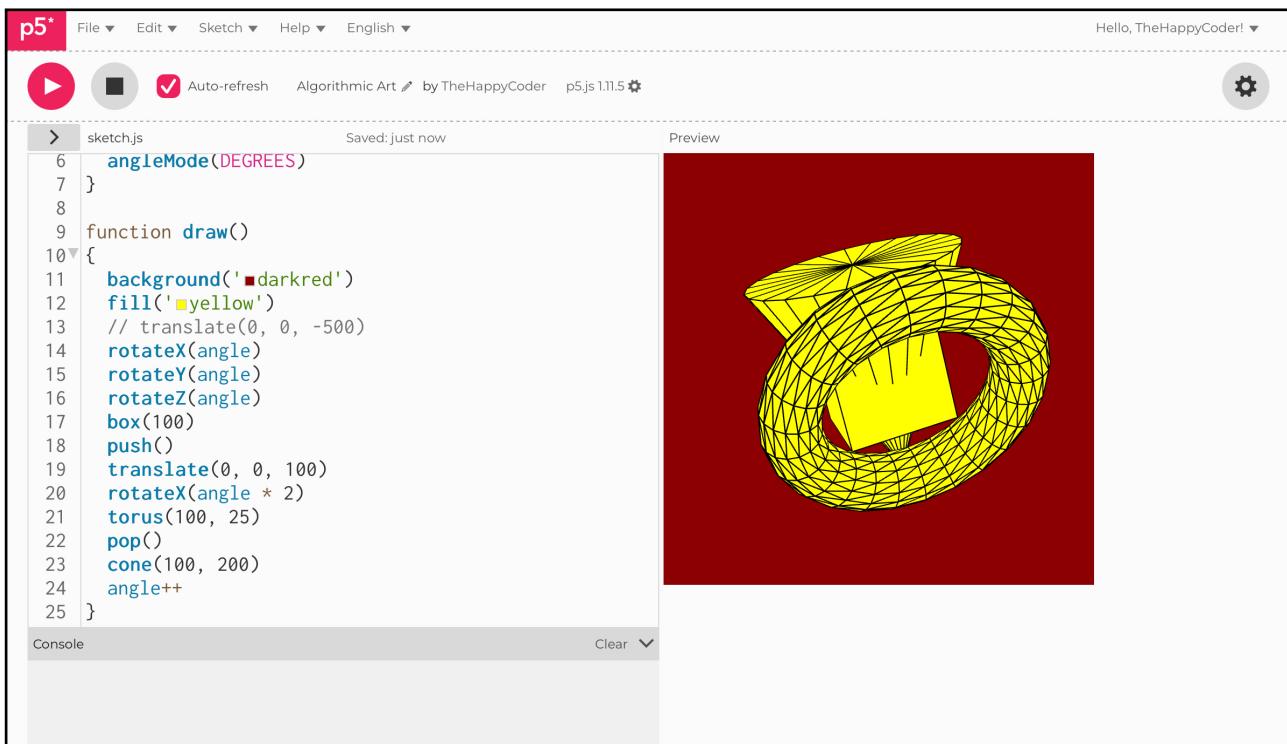
The torus moves independently of the other two shapes, and yet all three are still moving together. There is a lot of scope to do much more.



## Challenge

Just play and see what you can create.

Figure C1.21



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 gear icon. The main area has tabs for "sketch.js" and "Preview". The code in "sketch.js" is as follows:

```
6 angleMode(DEGREES)
7 }
8
9 function draw()
10{
11    background('darkred')
12    fill('yellow')
13    // translate(0, 0, -500)
14    rotateX(angle)
15    rotateY(angle)
16    rotateZ(angle)
17    box(100)
18    push()
19    translate(0, 0, 100)
20    rotateX(angle * 2)
21    torus(100, 25)
22    pop()
23    cone(100, 200)
24    angle++
25 }
```

The "Preview" window shows a 3D rendering of a yellow wireframe torus and a yellow cone on a dark red background. The torus is oriented vertically, and the cone is positioned in front of it.