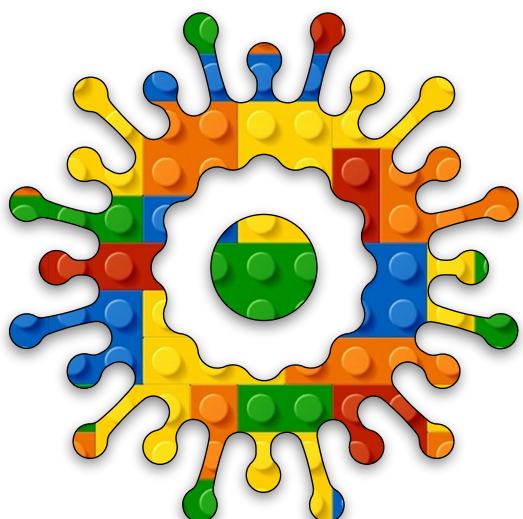


Creative  
Coding  
Module D  
Unit #2  
in another  
class





### Module D Unit #2 in another class

- Sketch D2.1 starting again
- Sketch D2.2 a single ball
- Sketch D2.3 adding a bit of gravity
- Sketch D2.4 creating a ball class
- Sketch D2.5 the constructor() function
- Sketch D2.6 the show() function
- Sketch D2.7 the move() function
- Sketch D2.8 can we have our ball back, please
- Sketch D2.9 many balls
- Sketch D2.10 pulling the balls out
- Sketch D2.11 random velocity
- Sketch D2.12 refactoring



## Introduction to another class

This is another simple example of using classes to create something. The previous unit introduced classes as well as comparing them with functions and objects. Here we add a few more features to create an explosion of balls cascading or raining down. We will draw on this when we create a fireworks display.



## Sketch D2.1 starting again

! starting a new sketch.  
A circle (ball) at ( **x**, **y** ) coordinates.

```
let x = 0
let y = 200

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

function draw()
{
    background('lightgrey')
    fill('yellow')
    circle(x, y, 50)
}
```

Figure D2.1

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 "Auto-refresh" with a checked checkbox, "Algorithmic Art" by "TheHappyCoder", and "p5.js 1.11.7". On the right side of the toolbar is a gear icon. Below the toolbar, the file name "sketch.js" is displayed along with the message "Saved: just now". To the right of the file name is a "Preview" button. The main area contains the p5.js code:

```
1 let x = 0
2 let y = 200
3
4 function setup()
5{
6  createCanvas(400, 400)
7}
8
9 function draw()
10{
11  background('lightgrey')
12  fill('yellow')
13  circle(x, y, 50)
14}
```

On the right, the preview window shows a light grey square with a single yellow circle centered in the middle.



## Sketch D2.2 a single ball

Now a ball moving across the canvas with a **x** velocity of **3** and a **y** velocity of **0**.

```
let x = 0
let y = 200
let x_velocity = 3
let y_velocity = 0

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

function draw()
{
    background('lightgrey')
    fill('yellow')
    circle(x, y, 50)
    x += x_velocity
    y += y_velocity
}
```

### Notes

Another naming convention is to use an underscore if splitting wordy and letters in a variable name.

Figure D2.2

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 gear icon. The main area has tabs for "sketch.js" and "Preview". The code editor contains the following JavaScript code:

```
1 let x = 0
2 let y = 200
3 let x_velocity = 3
4 let y_velocity = 0
5
6 function setup()
7 {
8   createCanvas(400, 400)
9 }
10
11 function draw()
12 {
13   background('lightgrey')
14   fill('yellow')
15   circle(x, y, 50)
16   x += x_velocity
17   y += y_velocity
18 }
```

The preview window on the right shows a light gray canvas with a single yellow circle centered at (200, 200). The circle moves to the right at a velocity of 3 pixels per frame.



## Sketch D2.3 adding a bit of gravity

We give the **y\_velocity** a bit of movement upwards (hence the negative value), but by adding a bit of **gravity**, it causes the ball to fall downwards to the ground after a while.

```
let x = 0
let y = 200
let x_velocity = 3
let y_velocity = -3
let gravity = 0.07

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

function draw()
{
    background('lightgrey')
    fill('yellow')
    circle(x, y, 50)
    x += x_velocity
    y += y_velocity
    y_velocity += gravity
}
```

### Notes

The ball goes up briefly and then starts to fall downwards once the velocity becomes positive.



## Challenges

1. Try different y velocities and gravity values.
2. Wind could be a variable in the x direction.

Figure D2.3

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 help. The title bar says "Hello, TheHappyCoder! ▾". The main area has tabs for "sketch.js" and "Preview". The code editor contains the following JavaScript code:

```
sketch.js
1 let x = 0
2 let y = 200
3 let x_velocity = 3
4 let y_velocity = -3
5 let gravity = 0.07
6
7 function setup()
8 {
9   createCanvas(400, 400)
10 }
11
12 function draw()
13 {
14   background('lightgrey')
15   fill('yellow')
16   circle(x, y, 50)
17   x += x_velocity
18   y += y_velocity
19   y_velocity += gravity
20 }
```

The preview window shows a light gray square canvas with a single yellow circle centered in it, representing a ball falling under gravity.



## Sketch D2.4 creating a ball class

We have just built the basic structure for the class. We have our **constructor()** function; we will also need a **show()** function and a **move()** function. I want to emphasise now that the names of these functions are made up; you can call each one of them whatever names you want. It is convention to have a constructor function, but the others can have other names and often do.

```
let x = 0
let y = 200
let x_velocity = 3
let y_velocity = -3
let gravity = 0.07

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

function draw()
{
    background('lightgrey')
    fill('yellow')
    circle(x, y, 50)
    x += x_velocity
    y += y_velocity
    y_velocity += gravity
}

class Ball
{
    constructor()
```

```
{  
}  
  
show()  
{  
}  
  
}  
  
move()  
{  
}  
}
```

## Notes

Nothing new will happen.

Figure D2.4

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 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, along with "Saved: just now". A gear icon is in the top right corner.

The main area contains the code for a class named "Ball". The code is as follows:

```
18 | }  
19 | }  
20 |  
21 | class Ball  
22 | {  
23 |   constructor()  
24 |   {  
25 |   }  
26 |   }  
27 |  
28 |   show()  
29 | {  
30 | }  
31 |  
32 |  
33 |   move()  
34 | {  
35 | }  
36 | }  
37 }
```

To the right of the code is a "Preview" window showing a single yellow circle centered on a gray background. Below the preview is a "Console" window which is currently empty. At the bottom of the editor, there are "Clear" and a dropdown menu.



## Sketch D2.5 the constructor() function

! remove the commented lines of code.

First, we will fill in the **constructor()** function using the variables highlighted. At the moment, you will get an error message if you try to run it now.

```
// let x = 0  
// let y = 200  
// let x_velocity = 3  
// let y_velocity = -3  
// let gravity = 0.07
```

```
function setup()  
{  
    createCanvas(400, 400)  
}  
  
function draw()  
{  
    background('lightgrey')  
    fill('yellow')  
    circle(x, y, 50)  
    x += x_velocity  
    y += y_velocity  
    y_velocity += gravity  
}  
  
class Ball  
{  
    constructor()  
{
```

```
    this.x = 0
    this.y = 200
    this.x_velocity = 3
    this.y_velocity = -3
    this.gravity = 0.07
}

show()
{
}

move()
{
}

}
```

## \*\*\*\*\* Notes

The `this._____` means it is referring to a specific ball, not necessarily to all of them. You should get an error message re: the `x` variable.



## Sketch D2.6 the show() function

! remove commented out lines of code.  
We can now fill in the show() function.

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

function draw()
{
    background('lightgrey')
    // fill('yellow')
    // circle(x, y, 50)
    x += x_velocity
    y += y_velocity
    y_velocity += gravity
}

class Ball
{
    constructor()
    {
        this.x = 0
        this.y = 200
        this.x_velocity = 3
        this.y_velocity = -3
        this.gravity = 0.07
    }
}

show()
```

```
{  
    fill('yellow')  
    circle(this.x, this.y, 50)  
}  
  
move()  
{  
}  
}  
}
```

## Notes

The `x` and `y` coordinates become `this.x` and `this.y` for each ball we create. Still an error message, but don't worry.



## Sketch D2.7 the move() function

! remove the commented lines of code.  
And finally, the **move()** function.

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

function draw()
{
    background('lightgrey')
    // x += x_velocity
    // y += y_velocity
    // y_velocity += gravity
}

class Ball
{
    constructor()
    {
        this.x = 0
        this.y = 200
        this.x_velocity = 3
        this.y_velocity = -3
        this.gravity = 0.07
    }

    show()
    {
        fill('yellow')
```

```
    circle(this.x, this.y, 50)
}

move()
{
    this.x += this.x_velocity
    this.y += this.y_velocity
    this.y_velocity += this.gravity
}
}
```

## Notes

You shouldn't get an error message now, but neither will you see anything. We need to create the ball next and call those functions.



## Sketch D2.8 can we have our ball back, please

We have finally got our ball back, phew!

```
let ball

function setup()
{
    createCanvas(400, 400)
    ball = new Ball()

}

function draw()
{
    background('lightgrey')
    ball.show()
    ball.move()

}

class Ball
{
    constructor()
    {
        this.x = 0
        this.y = 200
        this.x_velocity = 3
        this.y_velocity = -3
        this.gravity = 0.07
    }

    show()
    {
```

```
    fill('yellow')
    circle(this.x, this.y, 50)
}

move()
{
    this.x += this.x_velocity
    this.y += this.y_velocity
    this.y_velocity += this.gravity
}
}
```

## Notes

Everything should now be as before if all is well.

Figure D2.8

The screenshot shows the p5.js code editor interface. At the top, there are buttons for play/pause, refresh, and settings, along with the text "Hello, TheHappyCoder! ▾". The menu bar includes "File ▾", "Edit ▾", "Sketch ▾", "Help ▾", and "English ▾". The code editor window has a file named "sketch.js" with the status "Saved: just now". The code itself is as follows:

```
20 this.x = 0
21 this.y = 200
22 this.x_velocity = 3
23 this.y_velocity = -3
24 this.gravity = 0.07
25 }
26
27 show()
{
28   fill('yellow')
29   circle(this.x, this.y, 50)
30 }
31
32 move()
{
33   this.x += this.x_velocity
34   this.y += this.y_velocity
35   this.y_velocity += this.gravity
36 }
37
38 }
39 }
```

To the right of the code editor is a preview window showing a single yellow circle falling downwards from a height of 200 pixels. Below the code editor is a "Console" section with a "Clear" button.



## Sketch D2.9 many balls

! remove and replace `ball = new Ball()` with the `for()` loop.  
We can have more than one ball, so let's have `20` and for that we create an array to store them by pushing each new ball into that array, warning you that you will get an error message.

```
let ball
let balls = []
let number = 20

function setup()
{
    createCanvas(400, 400)
    for (let i = 0; i < number; i++)
    {
        balls.push(new Ball())
    }
}

function draw()
{
    background('lightgrey')
    ball.show()
    ball.move()
}

class Ball
{
    constructor()
    {
        this.x = 0
    }
}
```

```
    this.y = 200
    this.x_velocity = 3
    this.y_velocity = -3
    this.gravity = 0.07
}

show()
{
    fill('yellow')
    circle(this.x, this.y, 50)
}

move()
{
    this.x += this.x_velocity
    this.y += this.y_velocity
    this.y_velocity += this.gravity
}
}
```

## Notes

We have broken the sketch, but have no fear, we will fix that.



## Sketch D2.10 pulling the balls out

We need to pull each of the balls from the array and `show()` and `move()` each one. We use the `for...of` loop.

```
let ball
let balls = []
let number = 20

function setup()
{
    createCanvas(400, 400)
    for (let i = 0; i < number; i++)
    {
        balls.push(new Ball())
    }
}

function draw()
{
    background('lightgrey')
    for (ball of balls)
    {
        ball.show()
        ball.move()
    }
}

class Ball
{
```

```
    this.x = 0
    this.y = 200
    this.x_velocity = 3
    this.y_velocity = -3
    this.gravity = 0.07
}

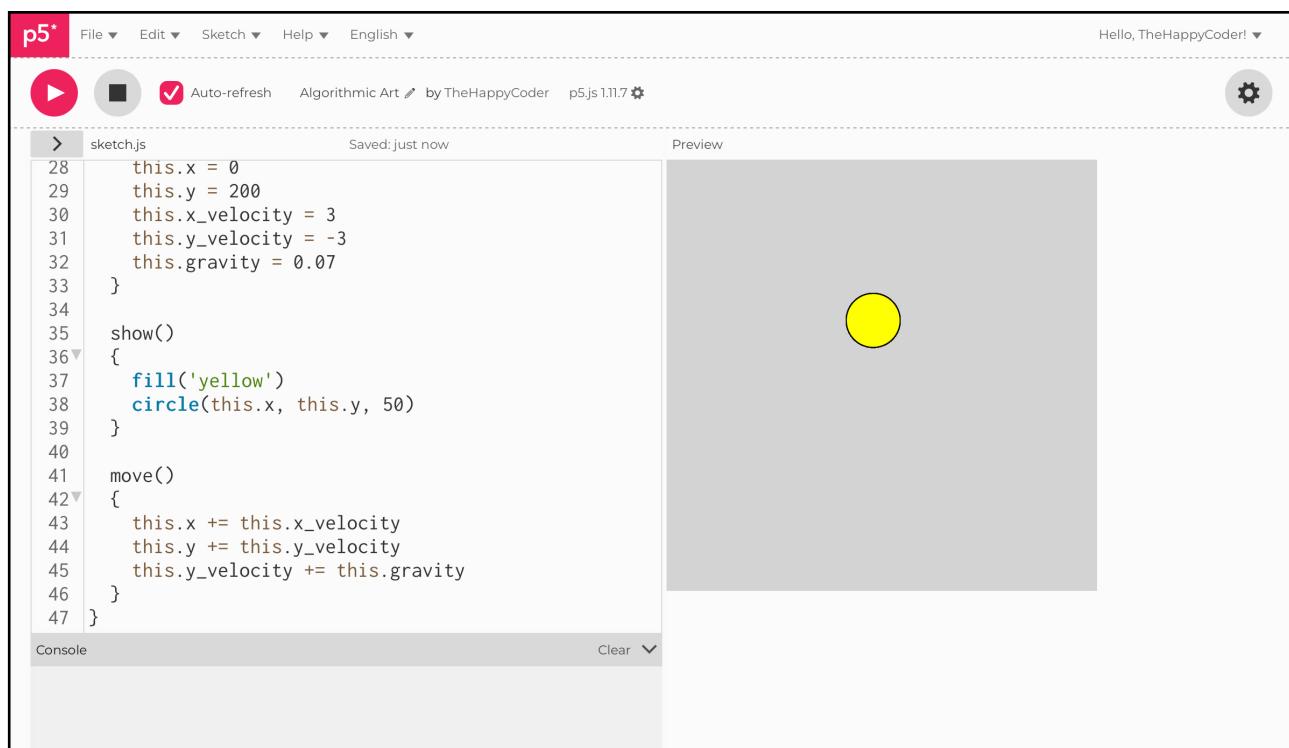
show()
{
    fill('yellow')
    circle(this.x, this.y, 50)
}

move()
{
    this.x += this.x_velocity
    this.y += this.y_velocity
    this.y_velocity += this.gravity
}
}
```

## Notes

We have fixed the sketch, but there were supposed to be **20** balls, not just one! To see them, we need to mix them up a bit; they are occupying the same space.

Figure D2.10



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 menu items: File ▾, Edit ▾, Sketch ▾, Help ▾, English ▾, and a user greeting: Hello, TheHappyCoder! ▾. Below the toolbar is a status bar with the file name sketch.js, a save message: Saved: just now, and a preview button.

The main area contains the sketch.js code:

```
sketch.js
28 this.x = 0
29 this.y = 200
30 this.x_velocity = 3
31 this.y_velocity = -3
32 this.gravity = 0.07
33 }
34
35 show()
36 {
37   fill('yellow')
38   circle(this.x, this.y, 50)
39 }
40
41 move()
42 {
43   this.x += this.x_velocity
44   this.y += this.y_velocity
45   this.y_velocity += this.gravity
46 }
47 }
```

To the right of the code is a preview window showing a single yellow circle centered on a gray background. The preview window has a close button in the top right corner.



## Sketch D2.11 random velocity

To mix it up, we can introduce some randomness to their velocities for both **x** and **y**.

```
let ball
let balls = []
let number = 20

function setup()
{
  createCanvas(400, 400)
  for (let i = 0; i < number; i++)
  {
    balls.push(new Ball())
  }
}

function draw()
{
  background('lightgrey')
  for (ball of balls)
  {
    ball.show()
    ball.move()
  }
}

class Ball
{
```

```
    this.x = 0
    this.y = 200
    this.x_velocity = 3 * random(1, 2)
    this.y_velocity = -3 * random(-1, 1)
    this.gravity = 0.07
}

show()
{
    fill('yellow')
    circle(this.x, this.y, 50)
}

move()
{
    this.x += this.x_velocity
    this.y += this.y_velocity
    this.y_velocity += this.gravity
}
}
```

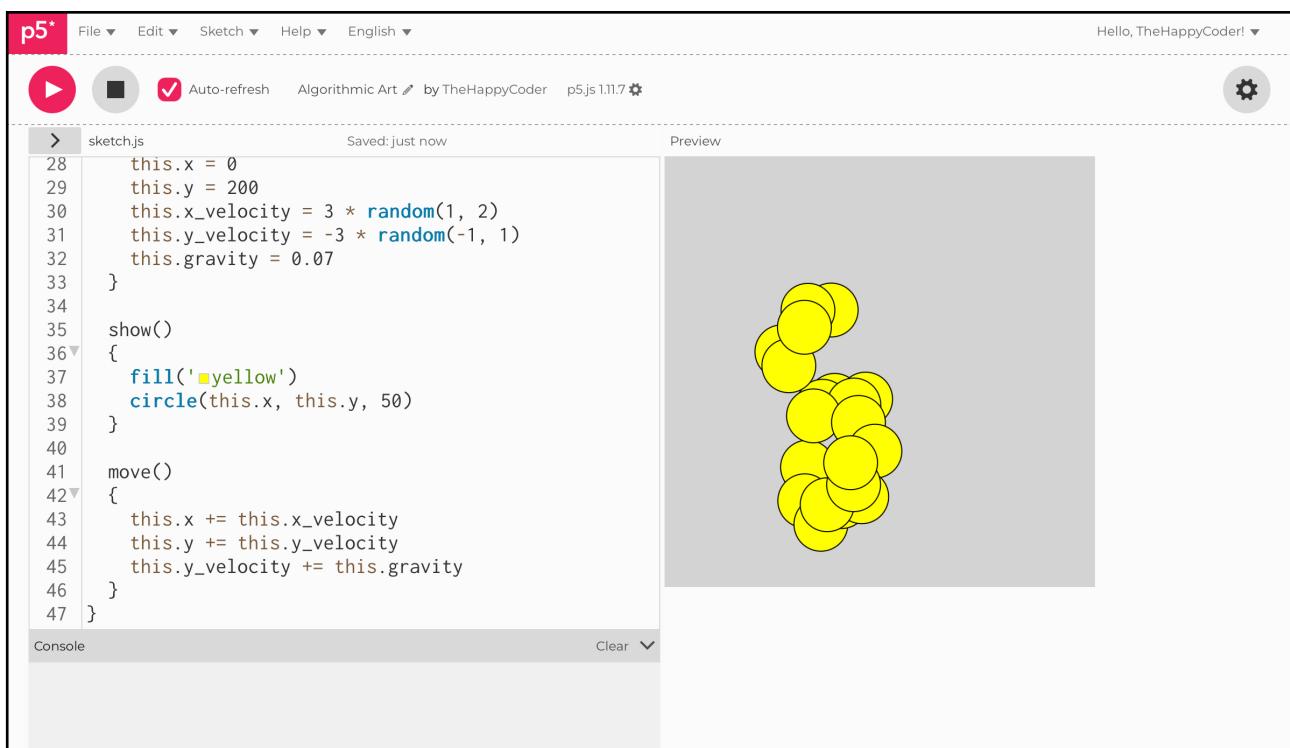
## Notes

They can now wander off in different directions.

## Challenge

Try other random values.

Figure D2.11



The screenshot shows the p5.js IDE interface. At the top, there are buttons for play/pause, stop, auto-refresh (which is checked), and settings. The title bar says "sketch.js" and "Saved: just now". The right side has a "Preview" window showing a cluster of yellow circles. The code editor contains the following JavaScript code:

```
28 this.x = 0
29 this.y = 200
30 this.x_velocity = 3 * random(1, 2)
31 this.y_velocity = -3 * random(-1, 1)
32 this.gravity = 0.07
33 }
34
35 show()
36 {
37   fill('yellow')
38   circle(this.x, this.y, 50)
39 }
40
41 move()
42 {
43   this.x += this.x_velocity
44   this.y += this.y_velocity
45   this.y_velocity += this.gravity
46 }
47 }
```

Below the code editor is a "Console" section which is currently empty. At the bottom right of the code editor is a "Clear" button.



## Sketch D2.12 refactoring

A bit of refactoring for a slightly different effect.

```
let ball
let balls = []
let number = 200

function setup()
{
  createCanvas(400, 400)
  for (let i = 0; i < number; i++)
  {
    balls.push(new Ball())
  }
}

function draw()
{
  background('lightgrey')
  for (ball of balls)
  {
    ball.show()
    ball.move()
  }
}

class Ball
{
  constructor()
  {
    this.x = 200
  }
}
```

```
    this.y = 200
    this.x_velocity = 3 * random(-1, 1)
    this.y_velocity = -3 * random(-1, 2)
    this.gravity = 0.07
}

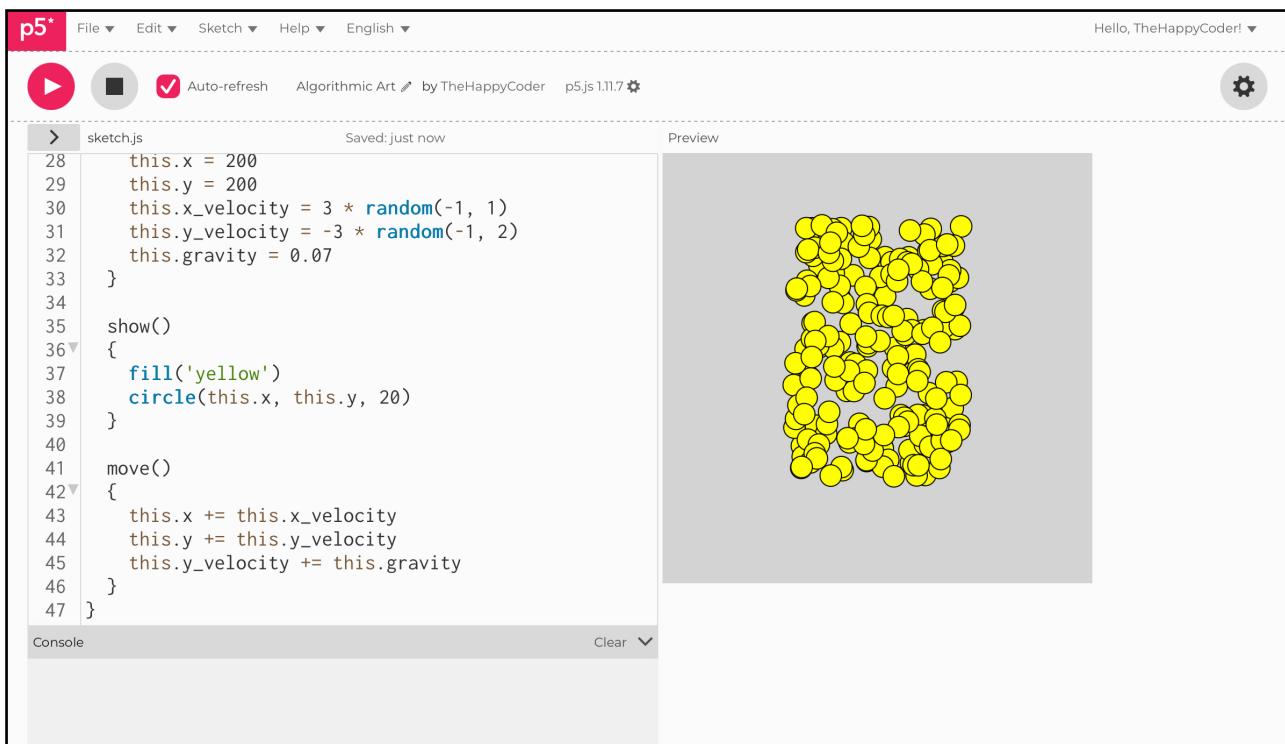
show()
{
    fill('yellow')
    circle(this.x, this.y, 20)
}

move()
{
    this.x += this.x_velocity
    this.y += this.y_velocity
    this.y_velocity += this.gravity
}
}
```

## Notes

We will have **200** smaller balls emitting from the centre; you could have something like a firework-type display.

Figure D2.12



The screenshot shows the p5.js code editor interface. At the top, there are navigation menus: 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:

```
sketch.js
28  this.x = 200
29  this.y = 200
30  this.x_velocity = 3 * random(-1, 1)
31  this.y_velocity = -3 * random(-1, 2)
32  this.gravity = 0.07
33 }
34
35 show()
36 {
37   fill('yellow')
38   circle(this.x, this.y, 20)
39 }
40
41 move()
42 {
43   this.x += this.x_velocity
44   this.y += this.y_velocity
45   this.y_velocity += this.gravity
46 }
47 }
```

The "Preview" window shows a cluster of yellow circles forming a stylized letter 'B'. The circles are yellow with black outlines, and they appear to be falling or moving downwards due to gravity.