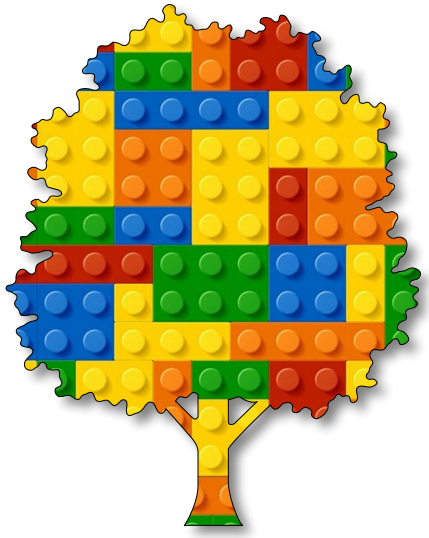
Algorithmic Art

Module E
Unit #5
fireworks display



Content

Module E Unit #5 fireworks display

Sketch E5.1	starting sketch
Sketch E5.2	index.html
Sketch E5.3	create vectors
Sketch E5.4	move() function
Sketch E5.5	creating a force
Sketch E5.6	adding the force
Sketch E5.7	draw the particle
Sketch E5.8	size and colour
Sketch E5.9	moving upwards
Sketch E5.10	upwards velocity
Sketch E5.11	gravity
Sketch E5.12	higher
Sketch E5.13	lots of fireworks
Sketch E5.14	explode
Sketch E5.15	remove redundant code
Sketch E5.16	cleaned up a bit
Sketch E5.17	from setup() to draw()
Sketch E5.18	not so many
Sketch E5.19	random height
Sketch E5.20	top of travel
Sketch E5.21	null
Sketch E5.22	boolean
Sketch E5.23	explode() function
Sketch E5.24	adding gravity
Sketch E5.25	two types of particle
Sketch E5.26	true
Sketch E5.27	random velocity
Sketch E5.28	slow down
Sketch E5.29	fade
Sketch E5.30	improve fade adjustment
Sketch E5.31	increase gravity reduce particles
Sketch E5.32	removing spent particles
Sketch E5.33	splice the faded
Sketch E5.34	it is done()
Sketch E5.35	sparks and rockets
Sketch E5.36	backwards array
Sketch E5.37	trails
Sketch E5.38	colour
Sketch E5.39	adding the colour in
Sketch E5.40	colorMode(RGB)

Introduction to fireworks display

Creating a firework display using particles. This builds on the previous unit covering simple particle systems. This is a bit all over the place in terms of simple tutorials; that doesn't mean that it is no good, rather it will need refactoring in the future. The final result is brilliant, but we do go round the houses a little bit.

We will be adding files and working from those separate files for some of the classes. To make matters clear, I will give each sketch a heading with the file it belongs to, and I will make clear references where we switch from one file to another. If you have never done this before, don't worry; you will very quickly get the hang of it.

We will be adding two more files: `particle.js` and `firework.js`. see below.

Sketch E5.1 starting sketch

❗ in sketch.js

Our starting sketch in the normal place.

sketch.js

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

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

Adding particle.js file

Add this file using the drop down menu (see module E unit #0 for more detail).

Figure 1: Add file called particle.js

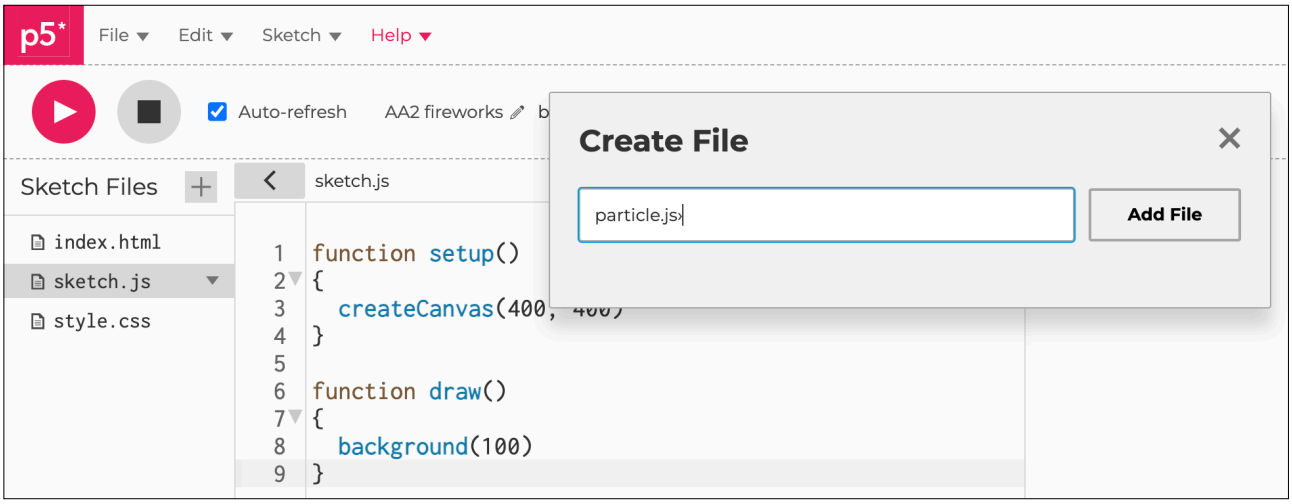
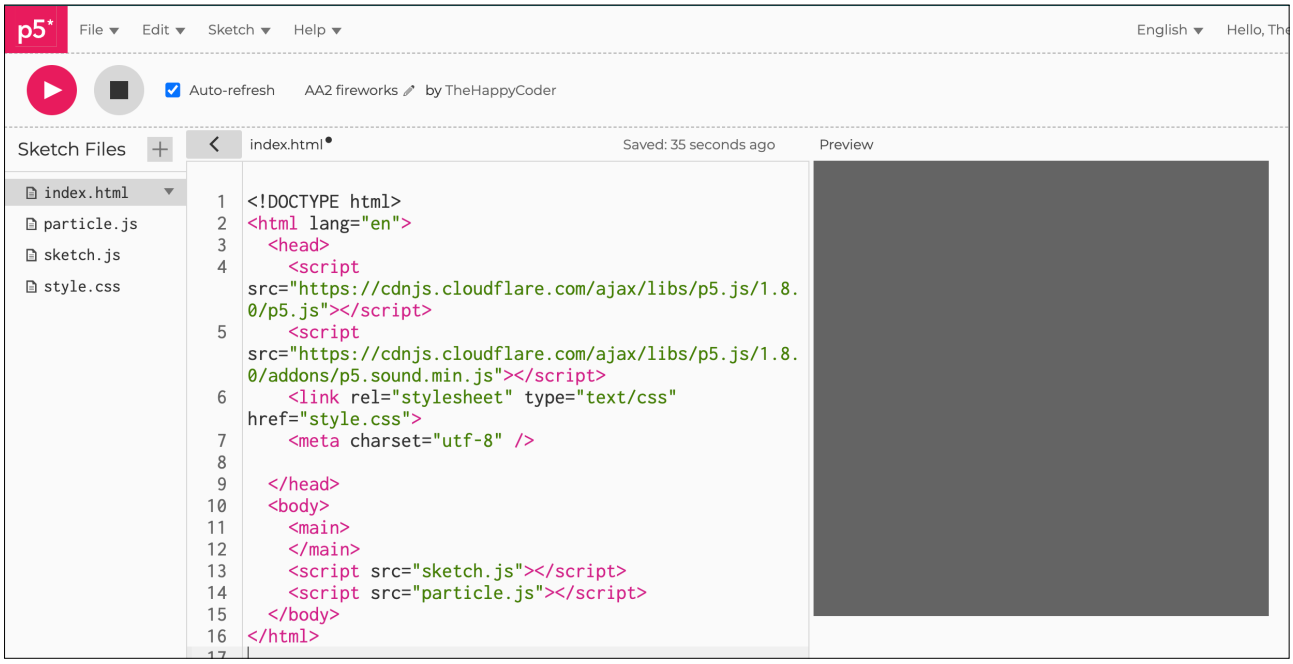


Figure 2: Add line of code into index.html



Sketch E5.2 index.html

❗ the index.html file

In the `index.html` file, we need to reference the particle.js file

index.html

```
<!DOCTYPE html>
<html lang="en">
  <head>
    <script src="https://cdnjs.cloudflare.com/ajax/libs/p5.js/1.8.0/p5.js"></
script>
    <script src="https://cdnjs.cloudflare.com/ajax/libs/p5.js/1.8.0/addons/
p5.sound.min.js"></script>
    <link rel="stylesheet" type="text/css" href="style.css">
    <meta charset="utf-8" />

  </head>
  <body>
    <main>
    </main>
    <script src="sketch.js"></script>
    <script src="particle.js"></script>
  </body>
</html>
```

Sketch E5.3 create vectors

❗ in the particle.js file

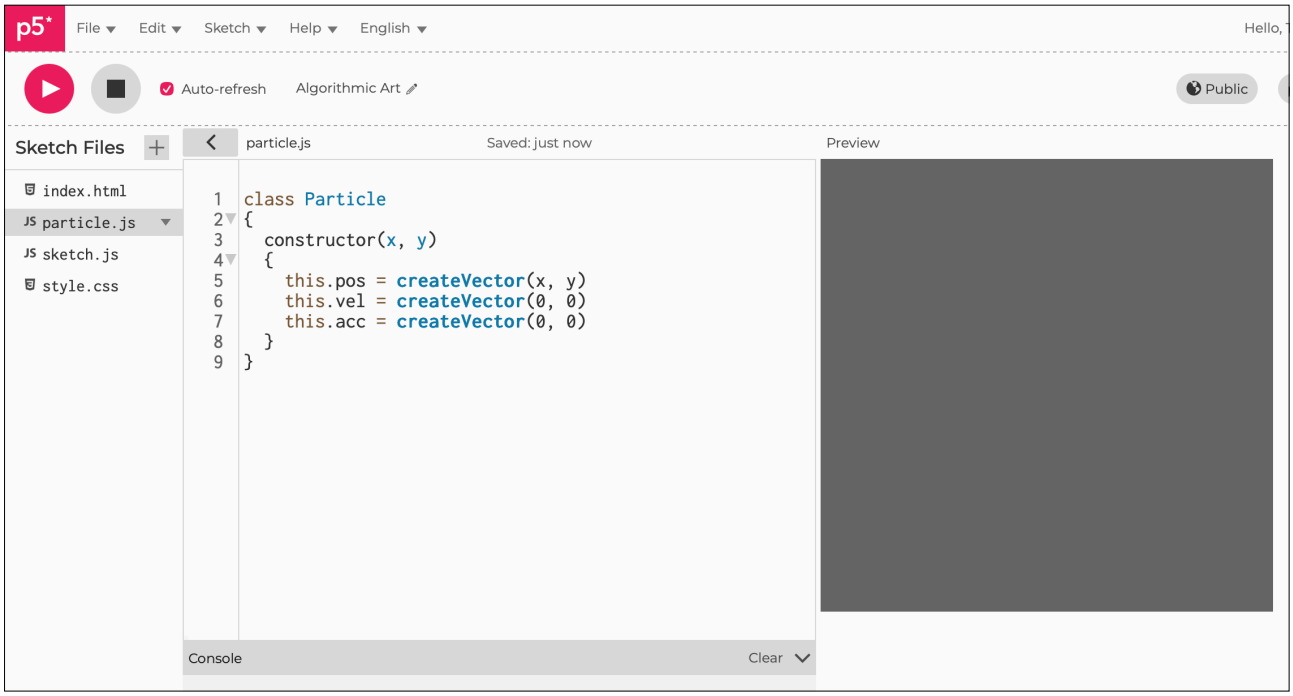
In a previous module, you were introduced to classes, so we create a particle class with a constructor that takes two variables (`x, y`) with three vectors:

1. position (`pos`)
2. velocity (`vel`)
3. acceleration (`acc`)

particle.js

```
class Particle
{
  constructor(x, y)
  {
    this.pos = createVector(x, y)
    this.vel = createVector(0, 0)
    this.acc = createVector(0, 0)
  }
}
```

Figure E5.3



Sketch E5.4 move() function

We create a `move()` function which adds the velocity to the position of the particle.

particle.js

```
class Particle
{
  constructor(x, y)
  {
    this.pos = createVector(x, y)
    this.vel = createVector(0, 0)
    this.acc = createVector(0, 0)
  }

  move()
  {
    this.pos.add(this.vel)
  }
}
```

Sketch E5.5 creating a force

To get the acceleration, we create a force. It will be fired upwards and fall under gravity; both are forces. We create a variable called force within which we will apply it, hence the function `applyForce()`. This is so the acceleration can accumulate.

From the basic equation `force = mass x acceleration`, but we can ignore mass, so `force = acceleration`.

particle.js

```
class Particle
{
  constructor(x, y)
  {
    this.pos = createVector(x, y)
    this.vel = createVector(0, 0)
    this.acc = createVector(0, 0)
  }

  applyForce(force)
  {
    this.acc.add(force)
  }

  move()
  {
    this.pos.add(this.vel)
  }
}
```

Sketch E5.6 adding the force

Adding this to the `move()` function. We multiply the acceleration (`acc`) by zero so that on each frame the acceleration is constant. (You can comment out this line to see what it does later.)

particle.js

```
class Particle
{
  constructor(x, y)
  {
    this.pos = createVector(x, y)
    this.vel = createVector(0, 0)
    this.acc = createVector(0, 0)
  }

  applyForce(force)
  {
    this.acc.add(force)
  }

  move()
  {
    this.vel.add(this.acc)
    this.pos.add(this.vel)
    this.acc.mult(0)
  }
}
```

Sketch E5.7 draw the particle

We now need to draw the particle, and so we create a `show()` function.

particle.js

```
class Particle
{
  constructor(x, y)
  {
    this.pos = createVector(x, y)
    this.vel = createVector(0, 0)
    this.acc = createVector(0, 0)
  }

  applyForce(force)
  {
    this.acc.add(force)
  }

  move()
  {
    this.vel.add(this.acc)
    this.pos.add(this.vel)
    this.acc.mult(0)
  }

  show()
  {
    point(this.pos.x, this.pos.y)
  }
}
```

Sketch E5.8 size and colour

❗ in the sketch.js file

You will notice that nothing happens when you run the sketch. So we need to move back to the main sketch.js. Here we give the particle a white colour (255) and a size (4). To test that this works, we add a variable called `firework` and use the `Particle class` to draw it.

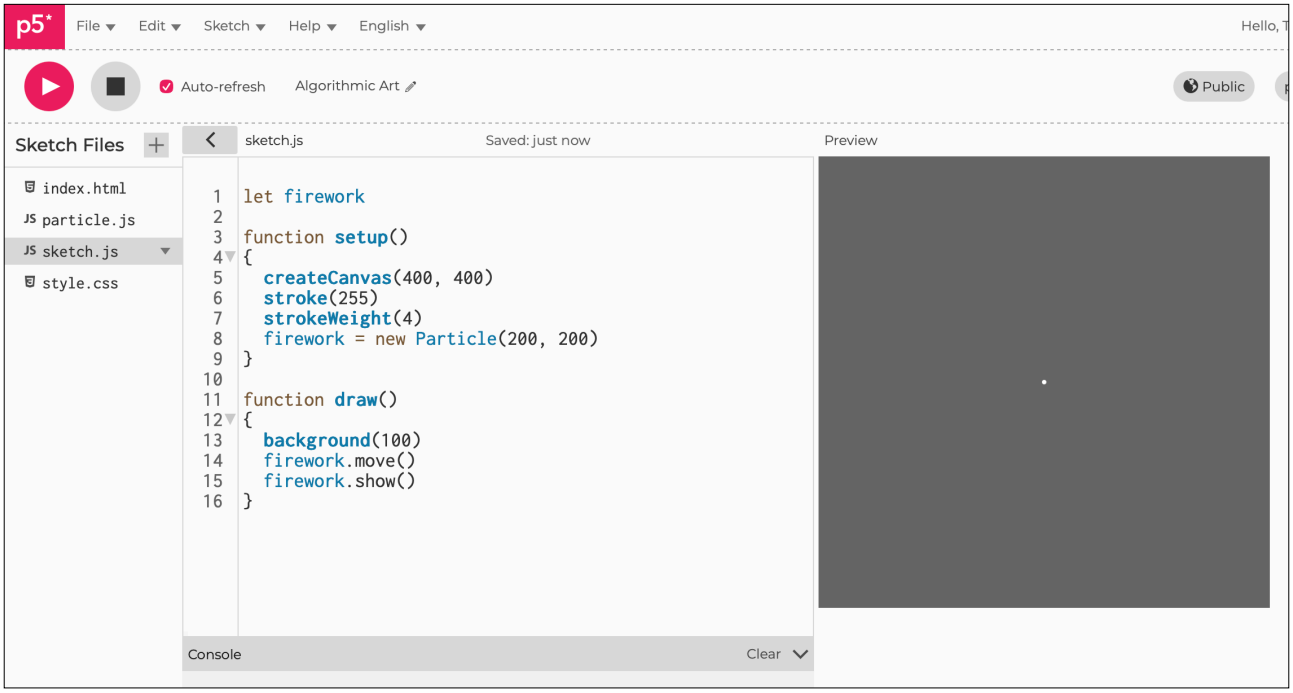
sketch.js

```
let firework

function setup()
{
  createCanvas(400, 400)
  stroke(255)
  strokeWeight(4)
  firework = new Particle(200, 200)
}

function draw()
{
  background(100)
  firework.move()
  firework.show()
}
```

🗒️ Notes

A new particle is drawn in the centre, something to see at last.

Figure E5.8


p5*
File
Edit
Sketch
Help
English
Auto-refresh
Algorithmic Art
Public
Sketch Files
index.html
particle.js
sketch.js
style.css
sketch.js
Saved: just now
Preview
let firework
function setup()
{
createCanvas(400, 400)
stroke(255)
strokeWeight(4)
firework = new Particle(200, 200)
}
function draw()
{
background(100)
firework.move()
firework.show()
}
Console
Clear

Sketch E5.9 moving upwards

Let's get it moving upwards from a new starting position on the ground at random.

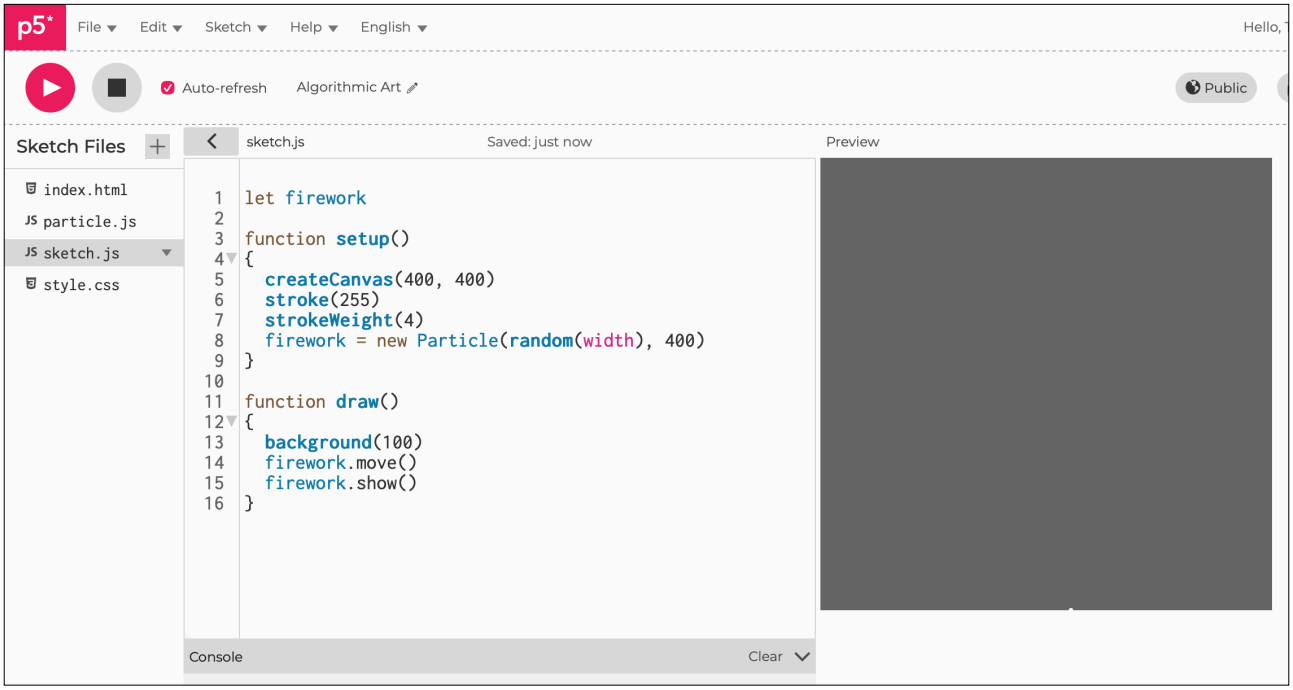
sketch.js

```
let firework

function setup()
{
  createCanvas(400, 400)
  stroke(255)
  strokeWeight(4)
  firework = new Particle(random(width), 400)
}

function draw()
{
  background(100)
  firework.move()
  firework.show()
}
```

Notes

You can just see it sitting at the bottom.

Figure E5.9


p5*
File
Edit
Sketch
Help
English
Auto-refresh
Algorithmic Art
Public
Sketch Files
index.html
particle.js
sketch.js
style.css
sketch.js
Saved: just now
Preview
let firework
function setup()
{
createCanvas(400, 400)
stroke(255)
strokeWeight(4)
firework = new Particle(random(width), 400)
}
function draw()
{
background(100)
firework.move()
firework.show()
}
Console
Clear

Sketch E5.10 upwards velocity

❗ in the particle.js file
In the `Particle class`, we need to give an upwards velocity of `-5`.

particle.js

```
class Particle
{
  constructor(x, y)
  {
    this.pos = createVector(x, y)
    this.vel = createVector(0, -5)
    this.acc = createVector(0, 0)
  }

  applyForce(force)
  {
    this.acc.add(force)
  }

  move()
  {
    this.vel.add(this.acc)
    this.pos.add(this.vel)
    this.acc.mult(0)
  }

  show()
  {
    point(this.pos.x, this.pos.y)
  }
}
```

🗒️ Notes

What you should see is the particle start at the bottom and move upwards and off the canvas.

Figure E5.10


p5*
File
Edit
Sketch
Help
English
Auto-refresh
Algorithmic Art
Public
Sketch Files
index.html
particle.js
sketch.js
style.css
particle.js
Saved: just now
Preview
{
this.pos = createVector(x, y)
this.vel = createVector(0, -5)
this.acc = createVector(0, 0)
}
applyForce(force)
{
this.acc.add(force)
}
move()
{
this.vel.add(this.acc)
this.pos.add(this.vel)
this.acc.mult(0)
}
show()
{
point(this.pos.x, this.pos.y)
}
}
Console
Clear

Sketch E5.11 gravity

❗ in sketch.js

Now let us add `gravity`, which is a force, and it points down (`0.1`).

sketch.js

```
let firework
let gravity

function setup()
{
  createCanvas(400, 400)
  gravity = createVector(0, 0.1)
  stroke(255)
  strokeWeight(4)
  firework = new Particle(random(width), 400)
}

function draw()
{
  background(100)
  firework.applyForce(gravity)
  firework.move()
  firework.show()
}
```

🗒️ Notes

You notice it doesn't go very high, so let's increase the vertical velocity to `8`.

Sketch E5.12 higher

❗ back to the particle.js file
Making it go a little higher.

particle.js

```
class Particle
{
  constructor(x, y)
  {
    this.pos = createVector(x, y)
    this.vel = createVector(0, -8)
    this.acc = createVector(0, 0)
  }

  applyForce(force)
  {
    this.acc.add(force)
  }

  move()
  {
    this.vel.add(this.acc)
    this.pos.add(this.vel)
    this.acc.mult(0)
  }

  show()
  {
    point(this.pos.x, this.pos.y)
  }
}
```

Sketch E5.13 lots of fireworks

❗ sketch.js file

At the moment, we have one firework shooting up, but we want lots of them (and then explode later on), so we create an array of fireworks.

sketch.js

```
let fireworks = []
let gravity

function setup()
{
  createCanvas(400, 400)
  gravity = createVector(0, 0.1)
  stroke(255)
  strokeWeight(4)
  firework = new Particle(random(width), 400)
}

function draw()
{
  background(100)
  firework.applyForce(gravity)
  firework.move()
  firework.show()
}
```

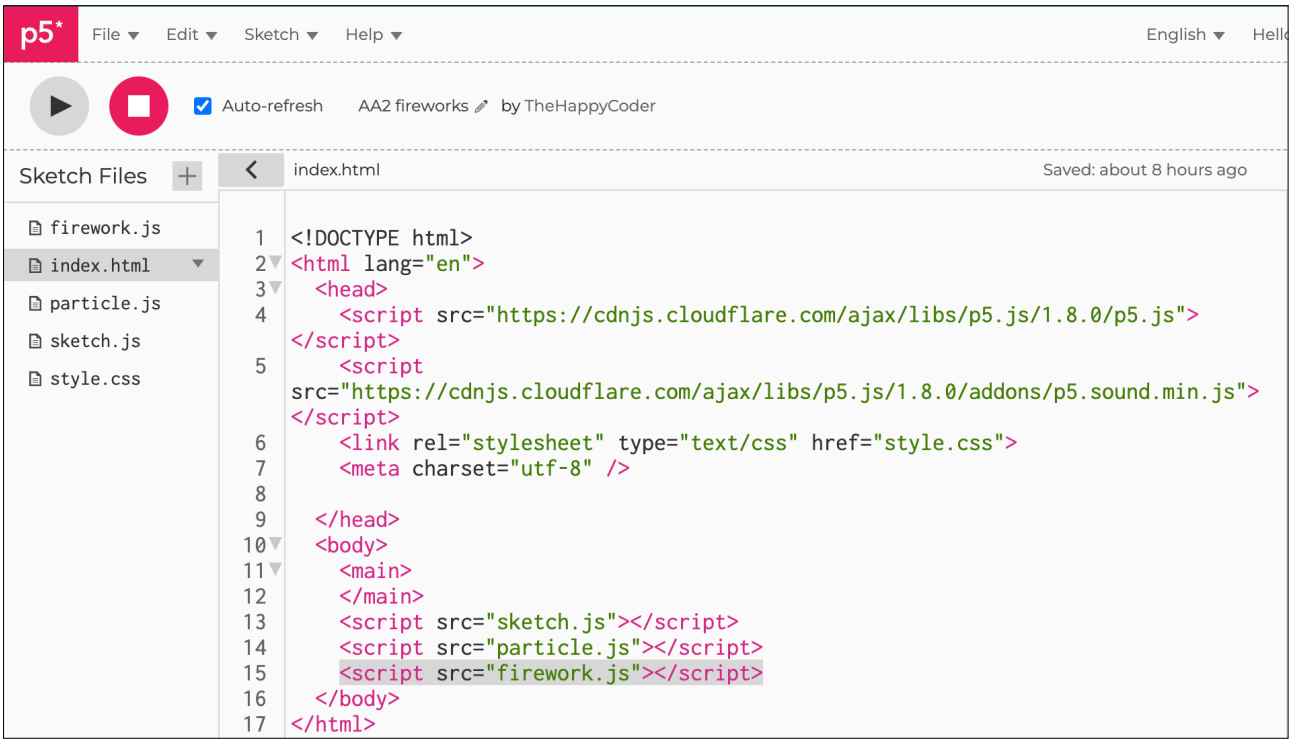
Notes

We still have the one particle going and returning to the ground.

Adding another file, firework.js

To facilitate this, we will create another file called `fireworks` and a corresponding class.

Figure 3: add another file called firework.js



Sketch E5.14 explode

❗ the firework.js file
We create a firework that is going to explode.

firework.js

```
class Firework
{
  constructor()
  {
    this.firework = new Particle(random(width), height)
  }

  move()
  {
    this.firework.applyForce(gravity)
    this.firework.move()
  }

  show()
  {
    this.firework.show()
  }
}
```

Sketch E5.15 remove redundant code

We need to reference these in the main sketch. No changes will be seen just yet.

❗ remove the redundant lines of code (commented //).

sketch.js

```
let fireworks = []
let gravity

function setup()
{
  createCanvas(400, 400)
  gravity = createVector(0, 0.1)
  fireworks.push(new Firework())
  stroke(255)
  strokeWeight(4)
  // firework = new Particle(random(width), 400)
}

function draw()
{
  background(100)
  for (let i = 0; i < fireworks.length; i++)
  {
    fireworks[i].move()
    fireworks[i].show()
  }
  // firework.applyForce(gravity)
  // firework.move()
  // firework.show()
}
```

Sketch E5.16 cleaned up a bit

Which should look like this now.

sketch.js

```
let fireworks = []
let gravity

function setup()
{
  createCanvas(400, 400)
  gravity = createVector(0, 0.1)
  fireworks.push(new Firework())
  stroke(255)
  strokeWeight(4)
}

function draw()
{
  background(100)
  for (let i = 0; i < fireworks.length; i++)
  {
    fireworks[i].move()
    fireworks[i].show()
  }
}
```

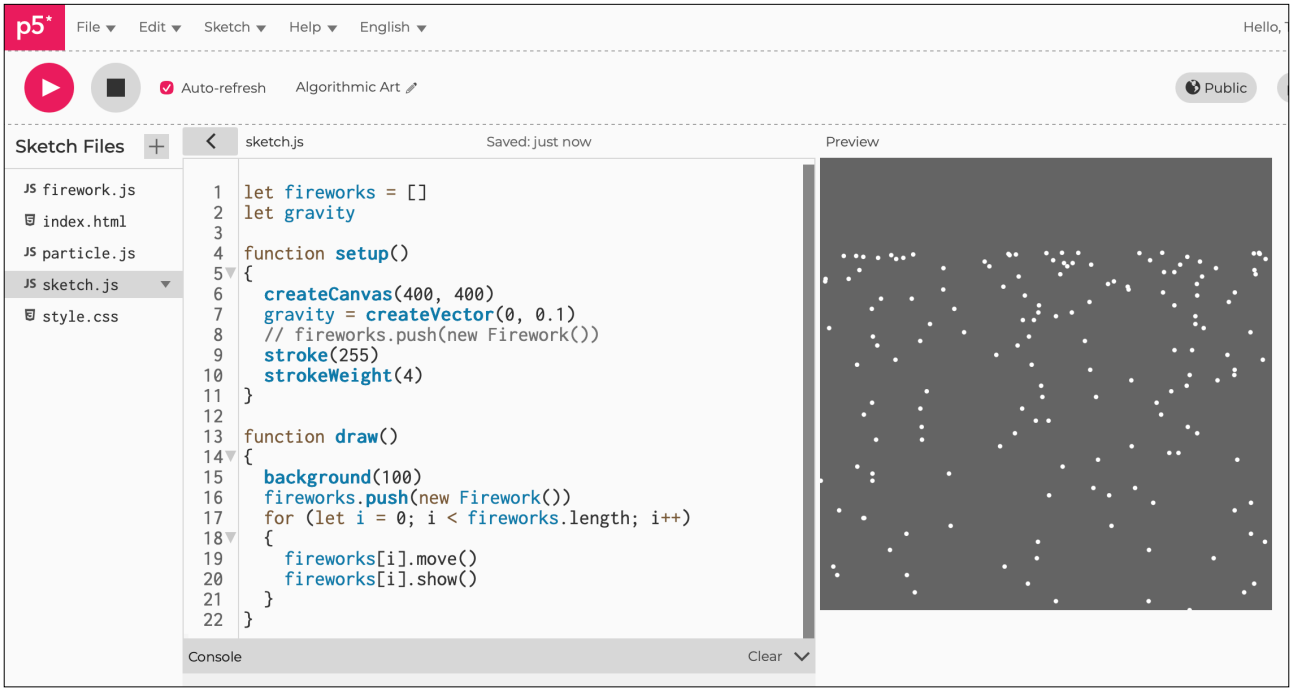
Sketch E5.17 from setup() to draw()

If we put the line of code creating an array of fireworks from `setup()` into `draw()`, we get lots of fireworks.

sketch.js

```
let fireworks = []
let gravity

function setup()
{
  createCanvas(400, 400)
  gravity = createVector(0, 0.1)
  // fireworks.push(new Firework())
  stroke(255)
  strokeWeight(4)
}

function draw()
{
  background(100)
  fireworks.push(new Firework())
  for (let i = 0; i < fireworks.length; i++)
  {
    fireworks[i].move()
    fireworks[i].show()
  }
}
```

Figure E5.17


p5*
File
Edit
Sketch
Help
English
Auto-refresh
Algorithmic Art
Public
Sketch Files
firework.js
index.html
particle.js
sketch.js
style.css
sketch.js
Saved: just now
Preview
let fireworks = []
let gravity
function setup()
{
createCanvas(400, 400)
gravity = createVector(0, 0.1)
// fireworks.push(new Firework())
stroke(255)
strokeWeight(4)
}
function draw()
{
background(100)
fireworks.push(new Firework())
for (let i = 0; i < fireworks.length; i++)
{
fireworks[i].move()
fireworks[i].show()
}
}
Console
Clear

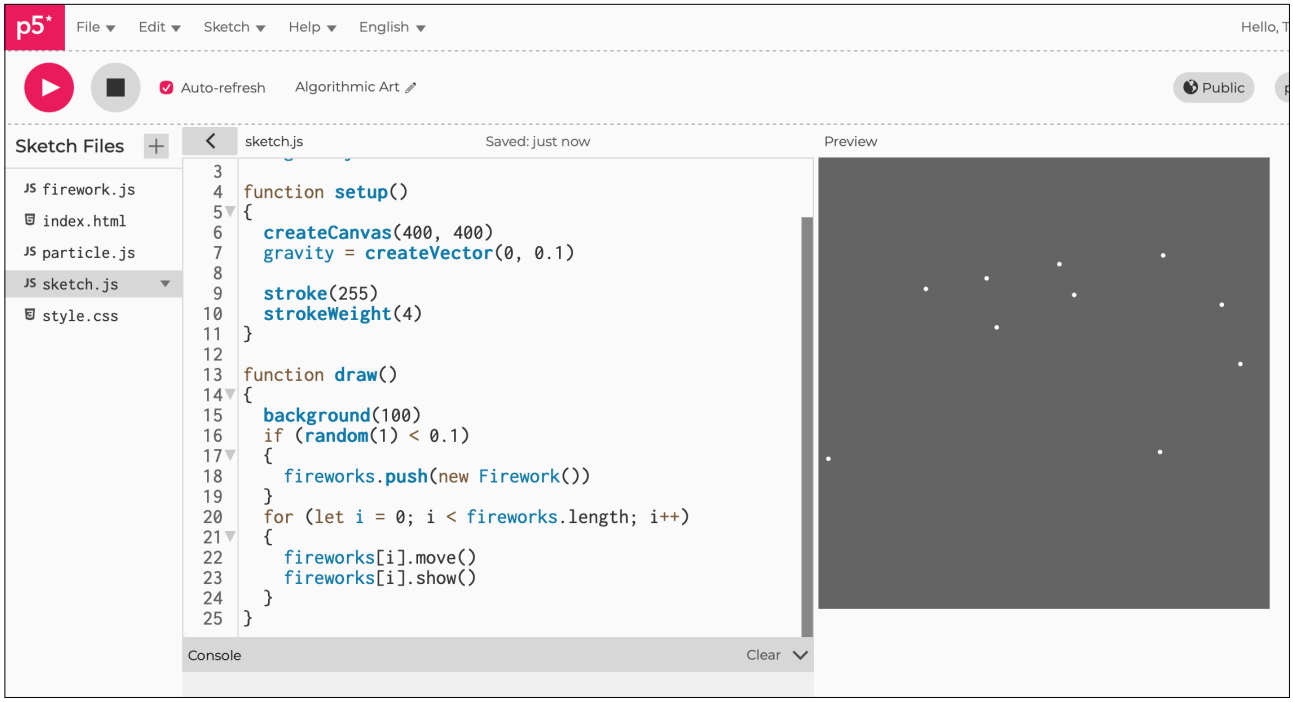
Sketch E5.18 not so many

We want more than one but not hundreds, so if we put the creation of a firework in the `draw()` function, we can limit how many it draws by using a random selection. In this case, every frame there is a `10%` chance of creating a firework.

sketch.js

```
let fireworks = []
let gravity

function setup()
{
  createCanvas(400, 400)
  gravity = createVector(0, 0.1)

  stroke(255)
  strokeWeight(4)
}

function draw()
{
  background(100)
  if (random(1) < 0.1)
  {
    fireworks.push(new Firework())
  }
  for (let i = 0; i < fireworks.length; i++)
  {
    fireworks[i].move()
    fireworks[i].show()
  }
}
```

Figure E5.18


p5*
File
Edit
Sketch
Help
English
Auto-refresh
Algorithmic Art
Public
Sketch Files
firework.js
index.html
particle.js
sketch.js
style.css
sketch.js
Saved: just now
Preview
3
4 function setup()
5 {
6 createCanvas(400, 400)
7 gravity = createVector(0, 0.1)
8
9 stroke(255)
10 strokeWeight(4)
11 }
12
13 function draw()
14 {
15 background(100)
16 if (random(1) < 0.1)
17 {
18 fireworks.push(new Firework())
19 }
20 for (let i = 0; i < fireworks.length; i++)
21 {
22 fireworks[i].move()
23 fireworks[i].show()
24 }
25 }
Console
Clear

Sketch E5.19 random height

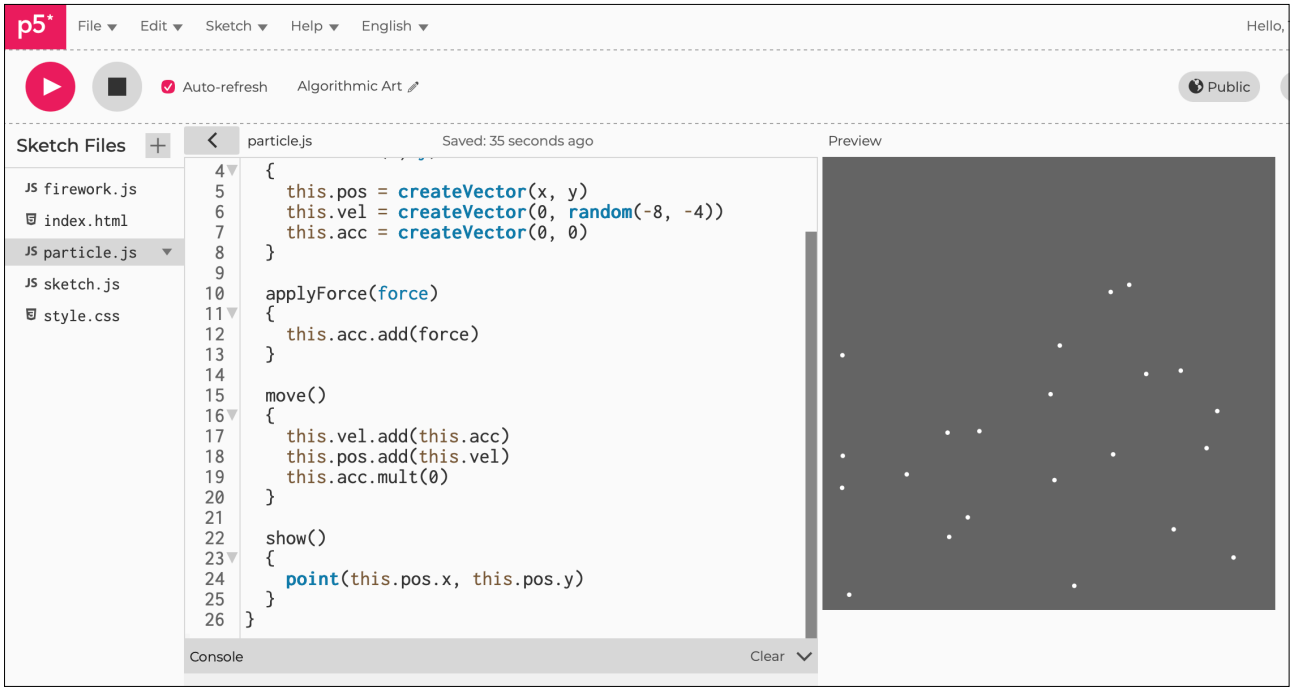
❗ the particle.js file

Let's randomise the height as well, so in particle.js we do the following. We can play with these later.

particle.js

```
class Particle
{
  constructor(x, y)
  {
    this.pos = createVector(x, y)
    this.vel = createVector(0, random(-8, -4))
    this.acc = createVector(0, 0)
  }

  applyForce(force)
  {
    this.acc.add(force)
  }

  move()
  {
    this.vel.add(this.acc)
    this.pos.add(this.vel)
    this.acc.mult(0)
  }

  show()
  {
    point(this.pos.x, this.pos.y)
  }
}
```

Figure E5.19


p5*
File
Edit
Sketch
Help
English
Hello,
Auto-refresh
Algorithmic Art
Public
Sketch Files
firework.js
index.html
particle.js
sketch.js
style.css
particle.js
Saved: 35 seconds ago
Preview
{
this.pos = createVector(x, y)
this.vel = createVector(0, random(-8, -4))
this.acc = createVector(0, 0)
}
applyForce(force)
{
this.acc.add(force)
}
move()
{
this.vel.add(this.acc)
this.pos.add(this.vel)
this.acc.mult(0)
}
show()
{
point(this.pos.x, this.pos.y)
}
}
Console
Clear

Sketch E5.20 top of travel

❗ the firework.js file

We want to explode the firework at the point it reaches the top of its travel. This is when the velocity is zero. As it goes up, it is negative, becomes zero, and then positive on the way down.

First, we only draw the firework if it exists by using the if statement (`this.firework`) in both `move()` and `show()`.

firework.js

```
class Firework
{
  constructor()
  {
    this.firework = new Particle(random(width), height)
  }

  move()
  {
    if (this.firework)
    {
      this.firework.applyForce(gravity)
      this.firework.move()
    }
  }

  show()
  {
    if (this.firework)
    {
      this.firework.show()
    }
  }
}
```

Sketch E5.21 null

Now, to make it null, which will make it disappear. This isn't permanent because we really want it to explode, but we will get to that in a moment.

firework.js

```
class Firework
{
  constructor()
  {
    this.firework = new Particle(random(width), height)
  }

  move()
  {
    if (this.firework)
    {
      this.firework.applyForce(gravity)
      this.firework.move()
      if (this.firework.vel.y >= 0)
      {
        this.firework = null
      }
    }
  }

  show()
  {
    if (this.firework)
    {
      this.firework.show()
      if (this.firework.vel.y >= 0)
      {
        this.firework = null
      }
    }
  }
}
```

🗒️ Notes

As it reaches its highest point, it then disappears.

Sketch E5.22 boolean

You get the idea, but we don't want to just remove them; we want to explode them. So we will change what happens at the top of the to a boolean expression. It will have the same effect. We will call the boolean variable `exploded`.

firework.js

```
class Firework
{
  constructor()
  {
    this.firework = new Particle(random(width), height)
    this.exploded = false
  }

  move()
  {
    if (!this.exploded)
    {
      this.firework.applyForce(gravity)
      this.firework.move()
      if (this.firework.vel.y >= 0)
      {
        this.exploded = true
      }
    }
  }

  show()
  {
    if (!this.exploded)
    {
      this.firework.show()
      if (this.firework.vel.y >= 0)
      {
        this.exploded = true
      }
    }
```

```
  }
}
```

🗒️ Notes

You could start with it being true, and then you would have to reverse all the logic in the rest of the sketch. It's a preference thing.

Sketch E5.23 explode() function

As we want to have the firework explode, we will create a function called `explode()`. We want it to explode when it reaches the very top (when it is null or true). In this function, we are going to make `100` particles.

❗ Removed the following lines of code:

```
if (this.firework.vel.y >= 0)
{
  this.exploded = true
}
```

firework.js

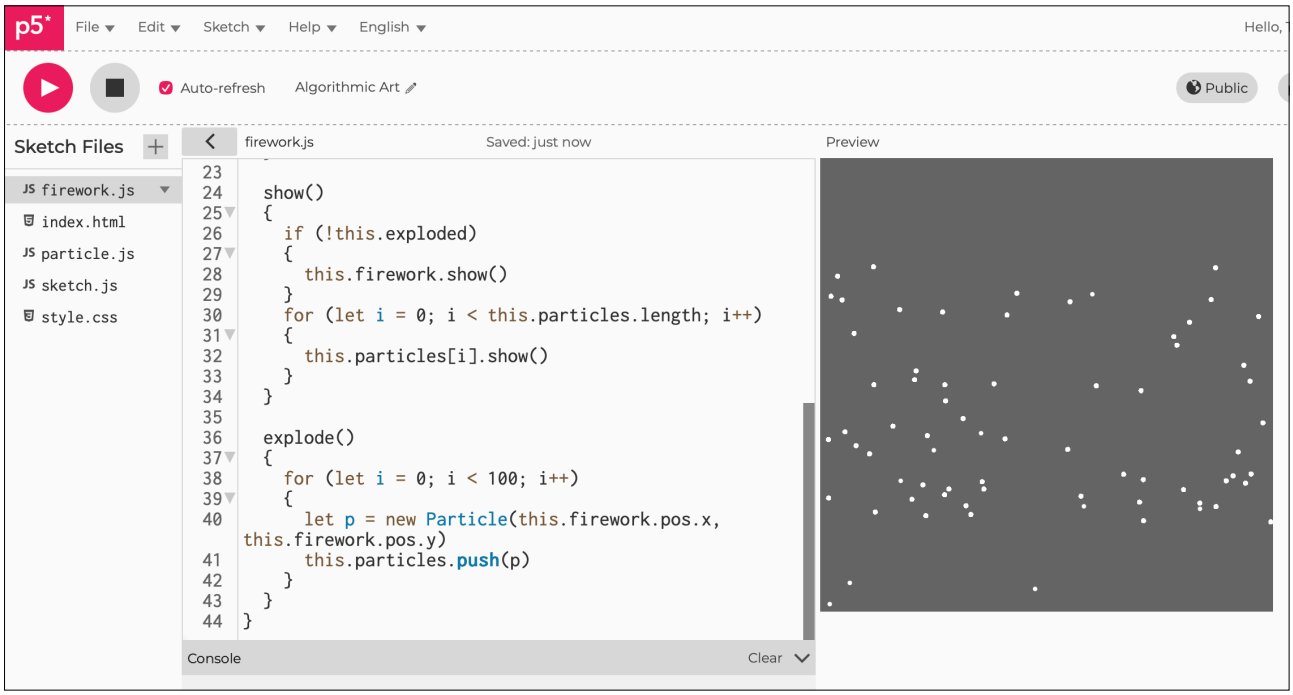
```
class Firework
{
  constructor()
  {
    this.firework = new Particle(random(width), height)
    this.exploded = false
    this.particles = []
  }

  move()
  {
    if (!this.exploded)
    {
      this.firework.applyForce(gravity)
      this.firework.move()
      if (this.firework.vel.y >= 0)
      {
        this.exploded = true
        this.explode()
      }
    }
  }

  show()
  {
    if (!this.exploded)
    {
```

```
      this.firework.show()
    }
    for (let i = 0; i < this.particles.length; i++)
    {
      this.particles[i].show()
    }
  }

  explode()
  {
    for (let i = 0; i < 100; i++)
    {
      let p = new Particle(this.firework.pos.x, this.firework.pos.y)
      this.particles.push(p)
    }
  }
}
```

🗒️ Notes

What you see is the particles going up, and then the 100 particles just stay put. We need to apply a force to those particles and move them.

Figure E5.23



Sketch E5.24 adding gravity

Applying gravity to the 100 particles.

firework.js

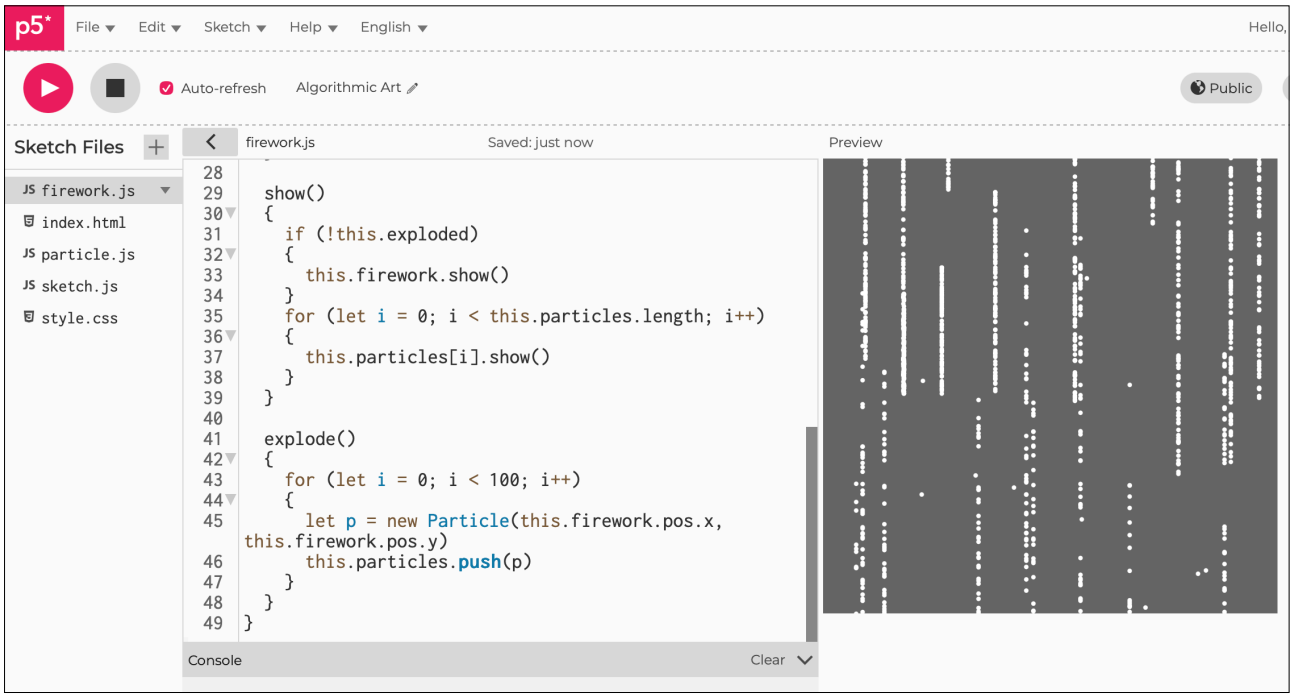
```
class Firework
{
  constructor()
  {
    this.firework = new Particle(random(width), height)
    this.exploded = false
    this.particles = []
  }

  move()
  {
    if (!this.exploded)
    {
      this.firework.applyForce(gravity)
      this.firework.move()
      if (this.firework.vel.y >= 0)
      {
        this.exploded = true
        this.explode()
      }
    }
    for (let i = 0; i < this.particles.length; i++)
    {
      this.particles[i].applyForce(gravity)
      this.particles[i].move()
    }
  }

  show()
  {
    if (!this.exploded)
    {
      this.firework.show()
```

```
    }
    for (let i = 0; i < this.particles.length; i++)
    {
      this.particles[i].show()
    }
  }

  explode()
  {
    for (let i = 0; i < 100; i++)
    {
      let p = new Particle(this.firework.pos.x, this.firework.pos.y)
      this.particles.push(p)
    }
  }
}
```

Notes

They are exploding but only moving vertically.

Figure E5.24



Sketch E5.25 two types of particle

❗ particle.js file

In particle.js, we add another argument. This is because we have two types of particle: one for the going-up bit (rocket firework) and the explode bit. This allows us to give different vector values to the two types of particle. `firework` becomes a boolean variable.

We use `p5.Vector.random2D()` so that we have a random vector in both the horizontal and the vertical.

particle.js

```
class Particle
{
  constructor(x, y, firework)
  {
    this.pos = createVector(x, y)
    if (firework)
    {
      this.vel = createVector(0, random(-8, -4))
    }
    else
    {
      this.vel = p5.Vector.random2D()
    }
    this.acc = createVector(0, 0)
  }

  applyForce(force)
  {
    this.acc.add(force)
  }

  move()
  {
    this.vel.add(this.acc)
    this.pos.add(this.vel)
    this.acc.mult(0)
  }
```

```
  show()
  {
    point(this.pos.x, this.pos.y)
  }
}
```

📝 Notes

You get them bubbling at the bottom of the canvas. We will raise them into the air.

Figure E5.25

Sketch E5.26 true

❗ firework.js

In the firework.js sketch, we need to add the boolean to be true for the particle.

firework.js

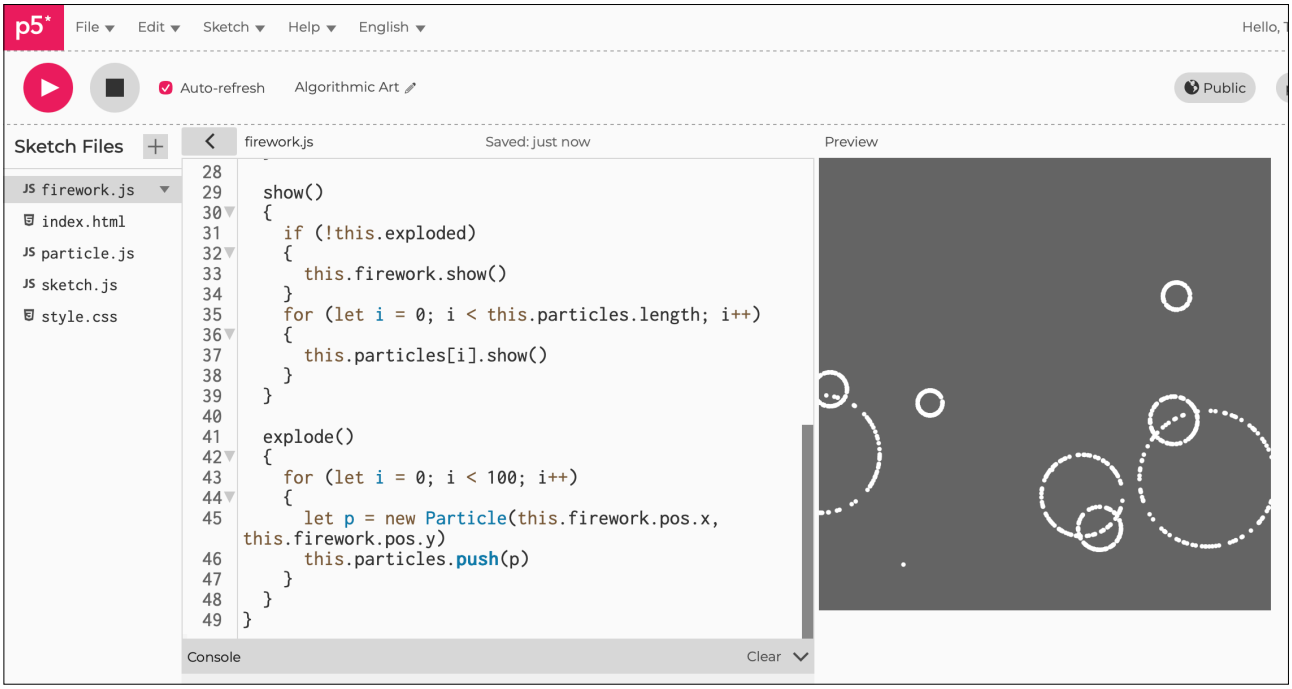
```
class Firework
{
  constructor()
  {
    this.firework = new Particle(random(width), height, true)
    this.exploded = false
    this.particles = []
  }

  move()
  {
    if (!this.exploded)
    {
      this.firework.applyForce(gravity)
      this.firework.move()
      if (this.firework.vel.y >= 0)
      {
        this.exploded = true
        this.explode()
      }
    }
    for (let i = 0; i < this.particles.length; i++)
    {
      this.particles[i].applyForce(gravity)
      this.particles[i].move()
    }
  }

  show()
  {
    if (!this.exploded)
    {
```

```
      this.firework.show()
    }
    for (let i = 0; i < this.particles.length; i++)
    {
      this.particles[i].show()
    }
  }

  explode()
  {
    for (let i = 0; i < 100; i++)
    {
      let p = new Particle(this.firework.pos.x, this.firework.pos.y)
      this.particles.push(p)
    }
  }
}
```

Notes

When you run it, you get something like this. Also, it starts running slower and slower as you create more and more particles. It looks rather nice as it is.

Figure E5.26



Sketch E5.27 random velocity

! particle.js

We have nice circles, but we want not just random direction but also random velocity.

particle.js

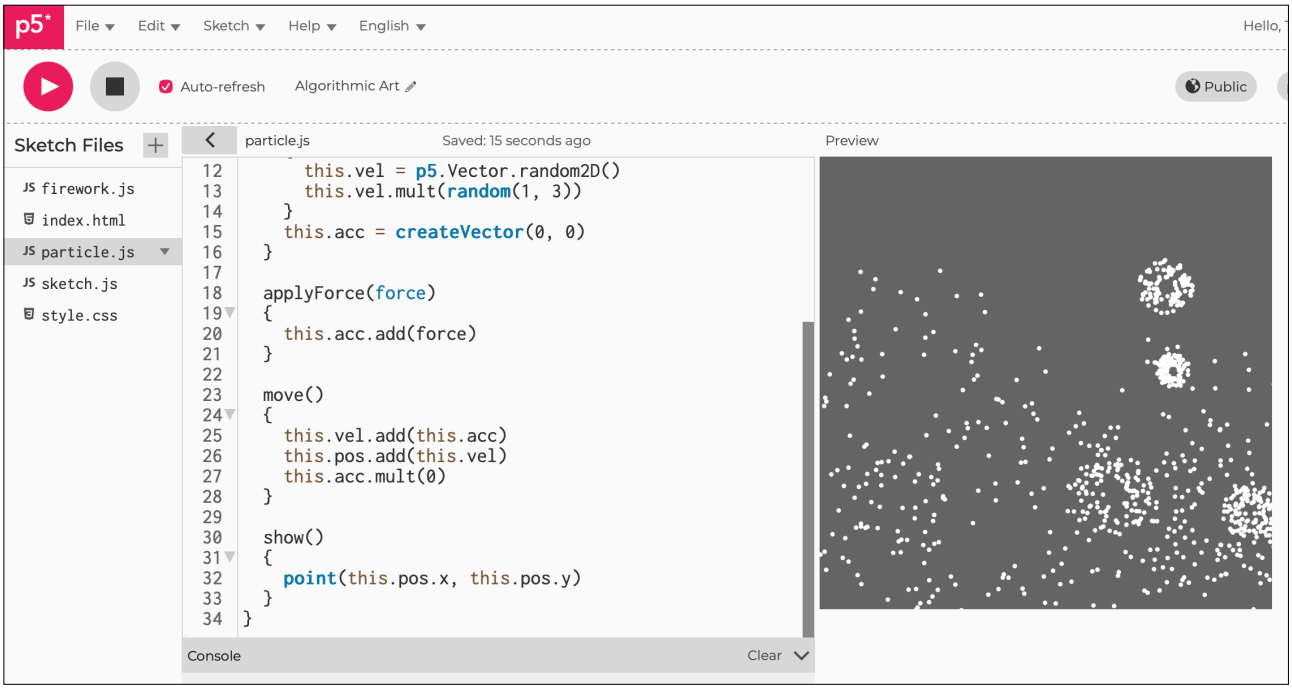
```
class Particle
{
  constructor(x, y, firework)
  {
    this.pos = createVector(x, y)
    if (firework)
    {
      this.vel = createVector(0, random(-8, -4))
    }
    else
    {
      this.vel = p5.Vector.random2D()
      this.vel.mult(random(1, 3))
    }
    this.acc = createVector(0, 0)
  }

  applyForce(force)
  {
    this.acc.add(force)
  }

  move()
  {
    this.vel.add(this.acc)
    this.pos.add(this.vel)
    this.acc.mult(0)
  }

  show()
  {
    point(this.pos.x, this.pos.y)
```

```
  }
}
```

Notes

They now explode a little bit more realistically.

Figure E5.27



Sketch E5.28 slow down

We want to keep track of those firework particles and cause them to not just fly away, also to slow down.

particle.js

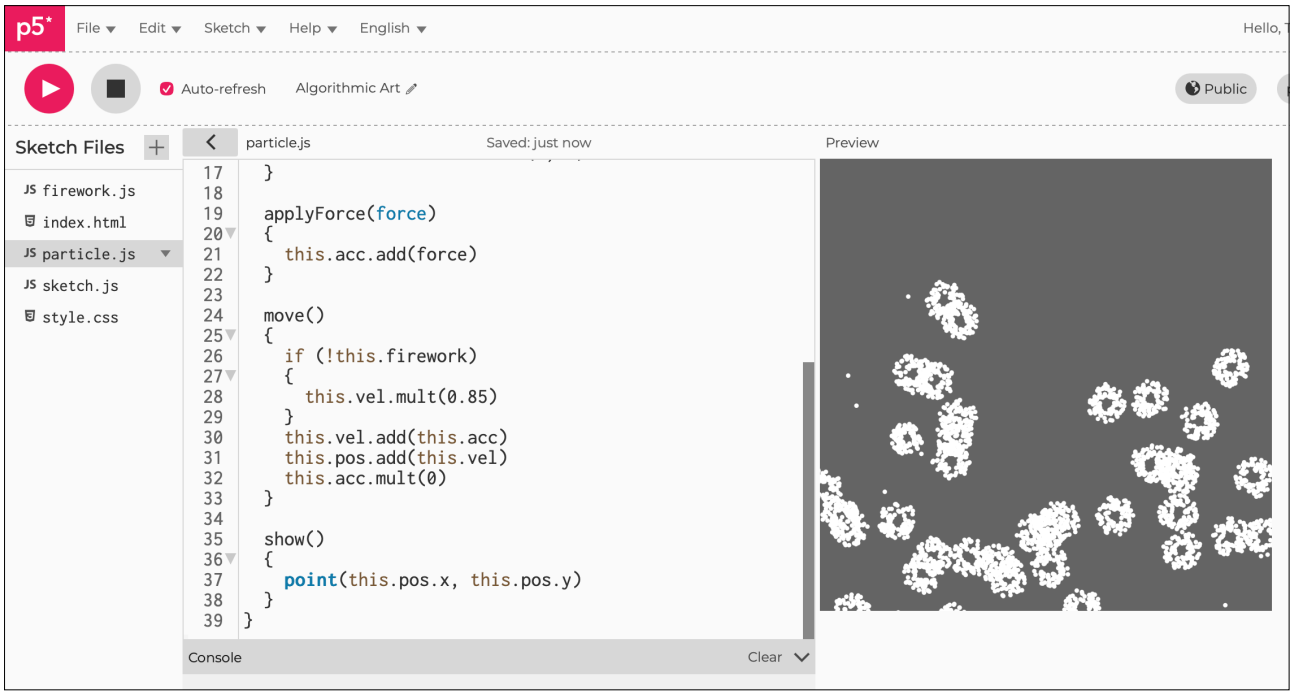
```
class Particle
{
  constructor(x, y, firework)
  {
    this.pos = createVector(x, y)
    this.firework = firework
    if (this.firework)
    {
      this.vel = createVector(0, random(-8, -4))
    }
    else
    {
      this.vel = p5.Vector.random2D()
      this.vel.mult(random(1, 3))
    }
    this.acc = createVector(0, 0)
  }

  applyForce(force)
  {
    this.acc.add(force)
  }

  move()
  {
    if (!this.firework)
    {
      this.vel.mult(0.85)
    }
    this.vel.add(this.acc)
    this.pos.add(this.vel)
    this.acc.mult(0)
```

```
  }

  show()
  {
    point(this.pos.x, this.pos.y)
  }
}
```

Notes

This has something of a bizarre effect, because the particles (or sparks) would fade quite soon after the initial explosion.

Figure E5.28



Sketch E5.29 fade

To get the fade effect, we create a new variable called `lifespan`. It will decrease by `4` on each frame and it will be drawn as a `stroke()` value.

particle.js

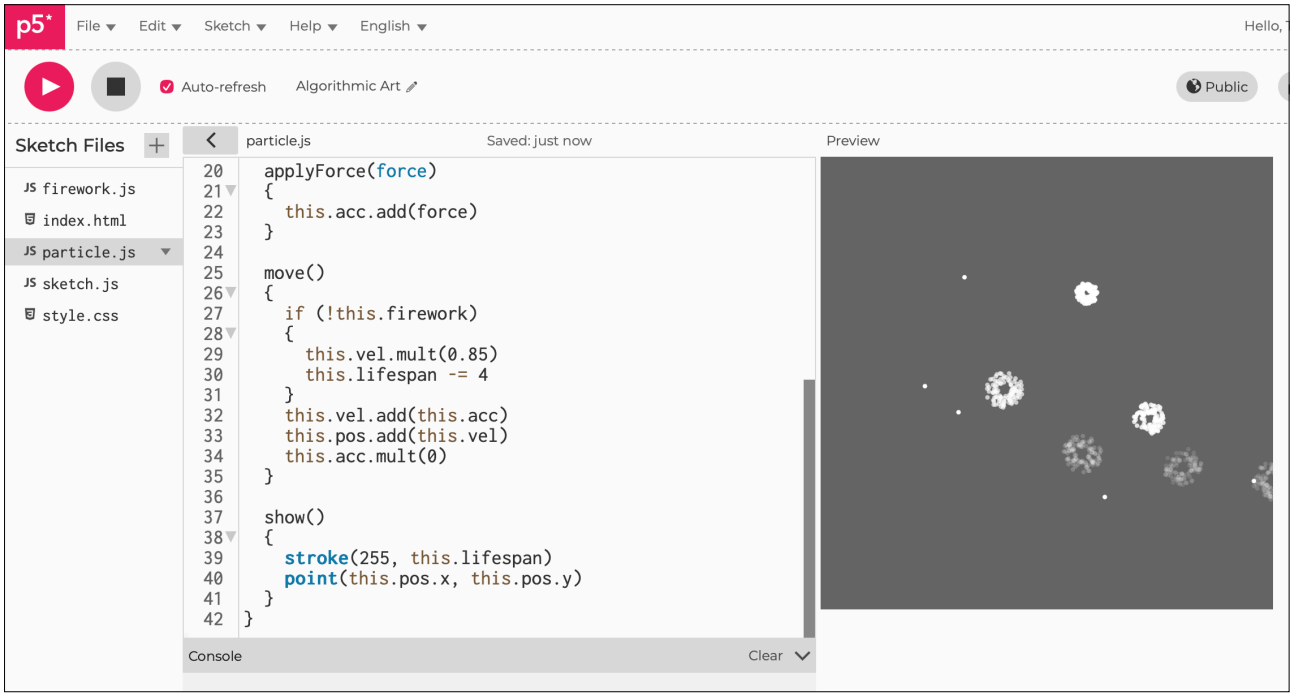
```
class Particle
{
  constructor(x, y, firework)
  {
    this.pos = createVector(x, y)
    this.firework = firework
    this.lifespan = 255
    if (this.firework)
    {
      this.vel = createVector(0, random(-8, -4))
    }
    else
    {
      this.vel = p5.Vector.random2D()
      this.vel.mult(random(1, 3))
    }
    this.acc = createVector(0, 0)
  }

  applyForce(force)
  {
    this.acc.add(force)
  }

  move()
  {
    if (!this.firework)
    {
      this.vel.mult(0.85)
      this.lifespan -= 4
    }
    this.vel.add(this.acc)
```

```
    this.pos.add(this.vel)
    this.acc.mult(0)
  }

  show()
  {
    stroke(255, this.lifespan)
    point(this.pos.x, this.pos.y)
  }
}
```

Notes

The `lifespan` variable becomes the `alpha` value and causes them to fade.

Figure E5.29



Sketch E5.30 improve fade adjustment

We want to have only the sparks fireworks fade, so a little bit of adjustment to a few parameters for velocity as well to give a more pleasing effect. You can play with these yourself.

particle.js

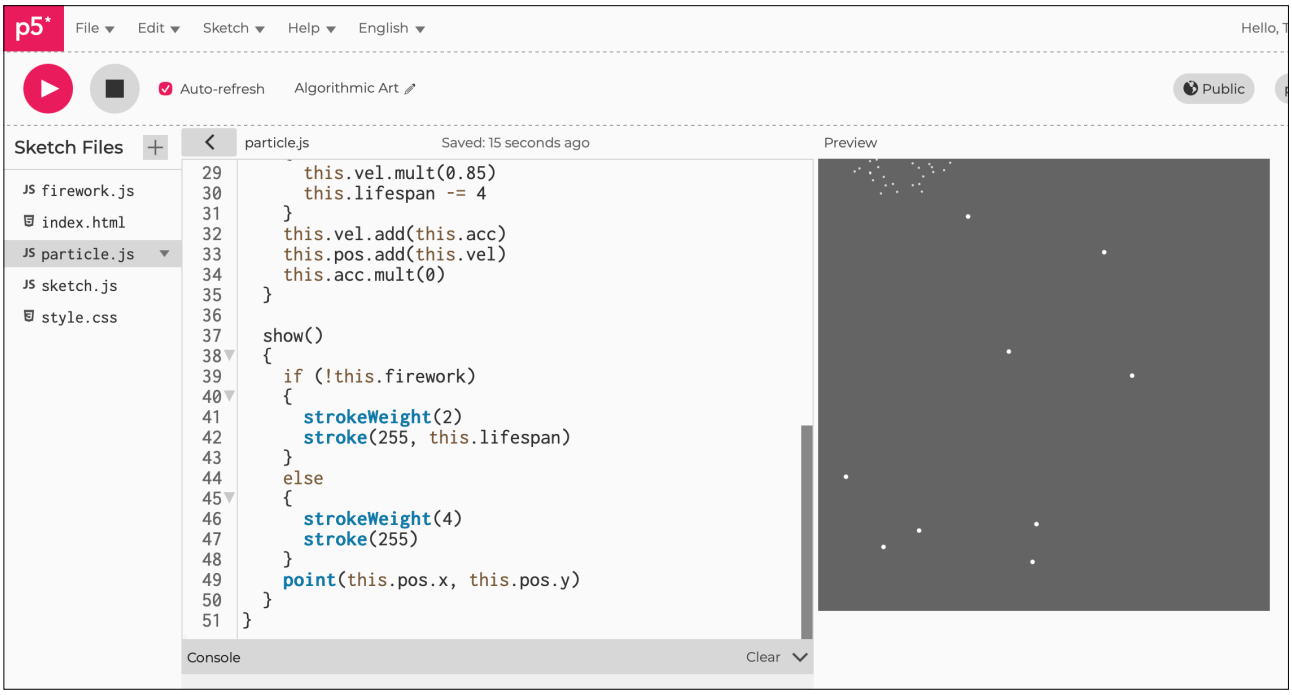
```
class Particle
{
  constructor(x, y, firework)
  {
    this.pos = createVector(x, y)
    this.firework = firework
    this.lifespan = 255
    if (this.firework)
    {
      this.vel = createVector(0, random(-12, -8))
    }
    else
    {
      this.vel = p5.Vector.random2D()
      this.vel.mult(random(2, 10))
    }
    this.acc = createVector(0, 0)
  }

  applyForce(force)
  {
    this.acc.add(force)
  }

  move()
  {
    if (!this.firework)
    {
      this.vel.mult(0.85)
      this.lifespan -= 4
    }
```

```
    this.vel.add(this.acc)
    this.pos.add(this.vel)
    this.acc.mult(0)
  }

  show()
  {
    if (!this.firework)
    {
      strokeWeight(2)
      stroke(255, this.lifespan)
    }
    else
    {
      strokeWeight(4)
      stroke(255)
    }
    point(this.pos.x, this.pos.y)
  }
}
```

🗒️ Notes

Definitely more like a firework display, but more to do.

Figure E5.30


p5*
File
Edit
Sketch
Help
English
Auto-refresh
Algorithmic Art
Public
Sketch Files
firework.js
index.html
particle.js
sketch.js
style.css
particle.js
Saved: 15 seconds ago
Preview
this.vel.mult(0.85)
this.lifespan -= 4
}
this.vel.add(this.acc)
this.pos.add(this.vel)
this.acc.mult(0)
}
show()
{
if (!this.firework)
{
strokeWeight(2)
stroke(255, this.lifespan)
}
else
{
strokeWeight(4)
stroke(255)
}
point(this.pos.x, this.pos.y)
}
}
Console
Clear

Sketch E5.31 increase gravity reduce particles

❗ sketch.js

We need to increase gravity a little more in sketch.js, and also reduce the number of particles being generated.

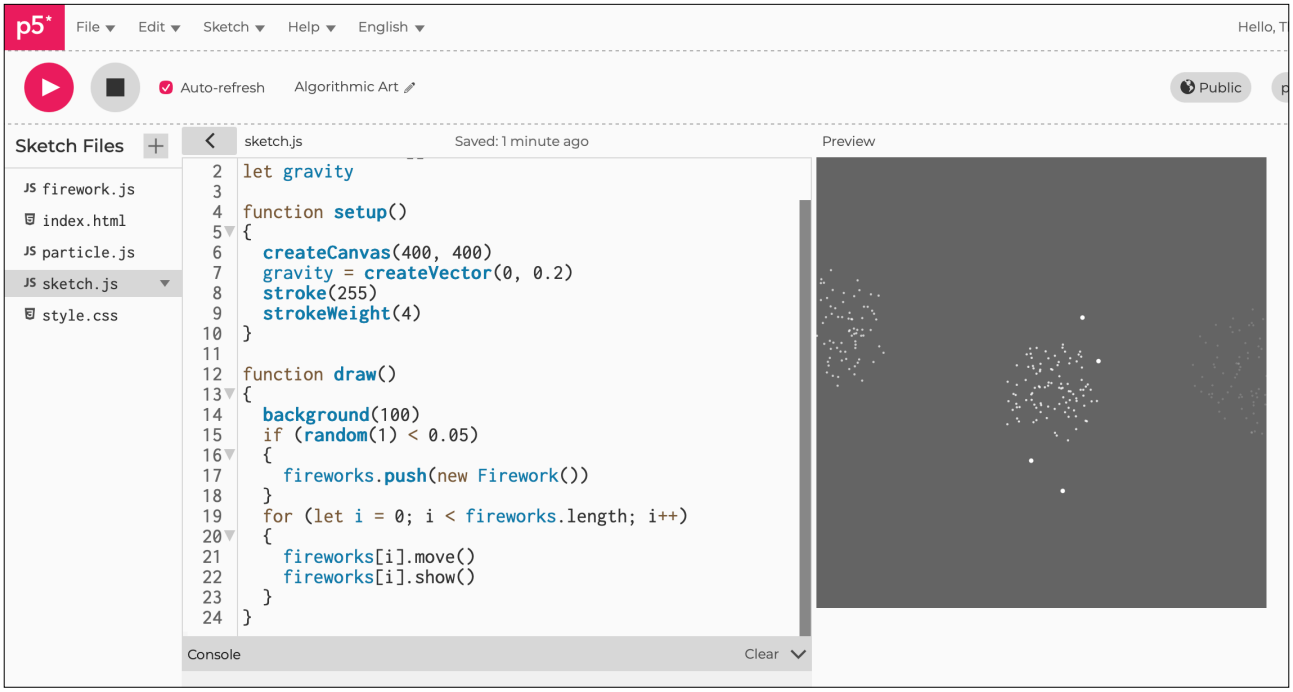
sketch.js

```
let fireworks = []
let gravity

function setup()
{
  createCanvas(400, 400)
  gravity = createVector(0, 0.2)
  stroke(255)
  strokeWeight(4)
}

function draw()
{
  background(100)
  if (random(1) < 0.05)
  {
    fireworks.push(new Firework())
  }
  for (let i = 0; i < fireworks.length; i++)
  {
    fireworks[i].move()
    fireworks[i].show()
  }
}
```

🗒 Notes

Still improving the effect.

Figure E5.31


p5*
File
Edit
Sketch
Help
English
Hello, T
Auto-refresh
Algorithmic Art
Public
Sketch Files
firework.js
index.html
particle.js
sketch.js
style.css
sketch.js
Saved: 1 minute ago
Preview
let gravity
function setup()
{
createCanvas(400, 400)
gravity = createVector(0, 0.2)
stroke(255)
strokeWeight(4)
}
function draw()
{
background(100)
if (random(1) < 0.05)
{
fireworks.push(new Firework())
}
for (let i = 0; i < fireworks.length; i++)
{
fireworks[i].move()
fireworks[i].show()
}
}
Console
Clear

Sketch E5.32 removing spent particles

❗ firework.js

We create lots of particles, but we never remove them from the array. This is why it starts to run very slowly after a short while. We have to delete those particles which have faded.

firework.js

```
class Firework
{
  constructor()
  {
    this.firework = new Particle(random(width), height, true)
    this.exploded = false
    this.particles = []
  }

  move()
  {
    if (!this.exploded)
    {
      this.firework.applyForce(gravity)
      this.firework.move()
      if (this.firework.vel.y >= 0)
      {
        this.exploded = true
        this.explode()
      }
    }
    for (let i = this.particles.length - 1; i >= 0; i--)
    {
      this.particles[i].applyForce(gravity)
      this.particles[i].move()
    }
  }

  show()
  {
```

```
    if (!this.exploded)
    {
      this.firework.show()
    }
    for (let i = 0; i < this.particles.length; i++)
    {
      this.particles[i].show()
    }
  }

  explode()
  {
    for (let i = 0; i < 100; i++)
    {
      let p = new Particle(this.firework.pos.x, this.firework.pos.y)
      this.particles.push(p)
    }
  }
}
```

🗒️ Notes

This should make no difference to the running of the sketch. If it does, then just check though that line of code.

Sketch E5.33 splice the faded

We need to splice, remove in this case the particle that is done (faded), done will be another function which we create in particle.js very shortly.

firework.js

```
class Firework
{
  constructor()
  {
    this.firework = new Particle(random(width), height, true)
    this.exploded = false
    this.particles = []
  }

  move()
  {
    if (!this.exploded)
    {
      this.firework.applyForce(gravity)
      this.firework.move()
      if (this.firework.vel.y >= 0)
      {
        this.exploded = true
        this.explode()
      }
    }
    for (let i = this.particles.length - 1; i >= 0; i--)
    {
      this.particles[i].applyForce(gravity)
      this.particles[i].move()
      if (this.particles[i].done())
      {
        this.particles.splice(i, 1)
      }
    }
  }
```

```
  show()
  {
    if (!this.exploded)
    {
      this.firework.show()
    }
    for (let i = 0; i < this.particles.length; i++)
    {
      this.particles[i].show()
    }
  }

  explode()
  {
    for (let i = 0; i < 100; i++)
    {
      let p = new Particle(this.firework.pos.x, this.firework.pos.y)
      this.particles.push(p)
    }
  }
}
```

🗒️ Notes

You will get an error message if you run the code.

Sketch E5.34 it is done()

! particle.js

Adding the `done()` function in particle.js, we use the boolean return as `true`, which means that it will splice (remove) that particle when it is done (`lifespan` is zero or less).

particle.js

```
class Particle
{
  constructor(x, y, firework)
  {
    this.pos = createVector(x, y)
    this.firework = firework
    this.lifespan = 255
    if (this.firework)
    {
      this.vel = createVector(0, random(-12, -8))
    }
    else
    {
      this.vel = p5.Vector.random2D()
      this.vel.mult(random(2, 10))
    }
    this.acc = createVector(0, 0)
  }

  applyForce(force)
  {
    this.acc.add(force)
  }

  move()
  {
    if (!this.firework)
    {
      this.vel.mult(0.85)
      this.lifespan -= 4
```

```
    }
    this.vel.add(this.acc)
    this.pos.add(this.vel)
    this.acc.mult(0)
  }

  done()
  {
    if (this.lifespan < 0)
    {
      return true
    }
    else
    {
      return false
    }
  }

  show()
  {
    if (!this.firework)
    {
      strokeWeight(2)
      stroke(255, this.lifespan)
    }
    else
    {
      strokeWeight(4)
      stroke(255)
    }
    point(this.pos.x, this.pos.y)
  }
}
```

🗒️ Notes

It should work fine now.

Sketch E5.35 sparks and rockets

❗ firework.js

We need to remove the rocket part of the firework now (we did the sparks). We want to delete the rocket firework when it has exploded and there are no more particles. First, we create a done() function for the firework.

firework.js

```
class Firework
{
  constructor()
  {
    this.firework = new Particle(random(width), height, true)
    this.exploded = false
    this.particles = []
  }

  done()
  {
    if (this.exploded && this.particles.length === 0)
    {
      return true
    }
    else
    {
      return false
    }
  }

  move()
  {
    if (!this.exploded)
    {
      this.firework.applyForce(gravity)
      this.firework.move()
      if (this.firework.vel.y >= 0)
      {
        this.exploded = true
```

```
        this.explode()
      }
    }
    for (let i = this.particles.length - 1; i >= 0; i--)
    {
      this.particles[i].applyForce(gravity)
      this.particles[i].move()
      if (this.particles[i].done())
      {
        this.particles.splice(i, 1)
      }
    }
  }

  show()
  {
    if (!this.exploded)
    {
      this.firework.show()
    }
    for (let i = 0; i < this.particles.length; i++)
    {
      this.particles[i].show()
    }
  }

  explode()
  {
    for (let i = 0; i < 100; i++)
    {
      let p = new Particle(this.firework.pos.x, this.firework.pos.y)
      this.particles.push(p)
    }
  }
}
```

🗒️ Notes

Looking good.

Sketch E5.36 backwards array

❗ sketch.js

Now we need to go to the sketch.js and firstly go through the array backwards and delete it (splice it). This may seem odd, but there is logic to this, just not immediately intuitive.

sketch.js

```
let fireworks = []
let gravity

function setup()
{
  createCanvas(400, 400)
  gravity = createVector(0, 0.2)
  stroke(255)
  strokeWeight(4)
}

function draw()
{
  background(100)
  if (random(1) < 0.05)
  {
    fireworks.push(new Firework())
  }
  for (let i = fireworks.length - 1; i >= 0; i--)
  {
    fireworks[i].move()
    fireworks[i].show()
    if (fireworks[i].done())
    {
      fireworks.splice(i, 1)
    }
  }
}
```

🗒️ Notes

You should see the same as before, but it should not slow down. You can check by adding this line of code at the end of `draw()`:

```
console.log(fireworks.length)
```

...and remove it once you are satisfied!

Sketch E5.37 trails

Let's add some trails to the fireworks.

sketch.js

```
let fireworks = []
let gravity

function setup()
{
  createCanvas(400, 400)
  gravity = createVector(0, 0.2)
  stroke(255)
  strokeWeight(4)
  background(0)
}

function draw()
{
  background(0, 25)
  if (random(1) < 0.05)
  {
    fireworks.push(new Firework())
  }
  for (let i = fireworks.length - 1; i >= 0; i--)
  {
    fireworks[i].move()
    fireworks[i].show()
    if (fireworks[i].done())
    {
      fireworks.splice(i, 1)
    }
  }
}
```

Notes

We now have something resembling a fireworks display.

Figure E5.37


p5*
File
Edit
Sketch
Help
English
Auto-refresh
Algorithmic Art
Public
Sketch Files
firework.js
index.html
particle.js
sketch.js
style.css
sketch.js
Saved: just now
Preview
gravity = createVector(0, 0.2)
stroke(255)
strokeWeight(4)
background(0)
}
function draw()
{
background(0, 25)
if (random(1) < 0.05)
{
fireworks.push(new Firework())
}
for (let i = fireworks.length - 1; i >= 0; i--)
{
fireworks[i].move()
fireworks[i].show()
if (fireworks[i].done())
{
fireworks.splice(i, 1)
}
}
}
Console
Clear

Sketch E5.38 colour

! firework.js

We are going to add colour with HSB rather than RGB (just a bit nicer). In the firework.js sketch, we introduce a variable called hu, which is the first argument for the colourMode(HSB). It is going to generate a random colour.

firework.js

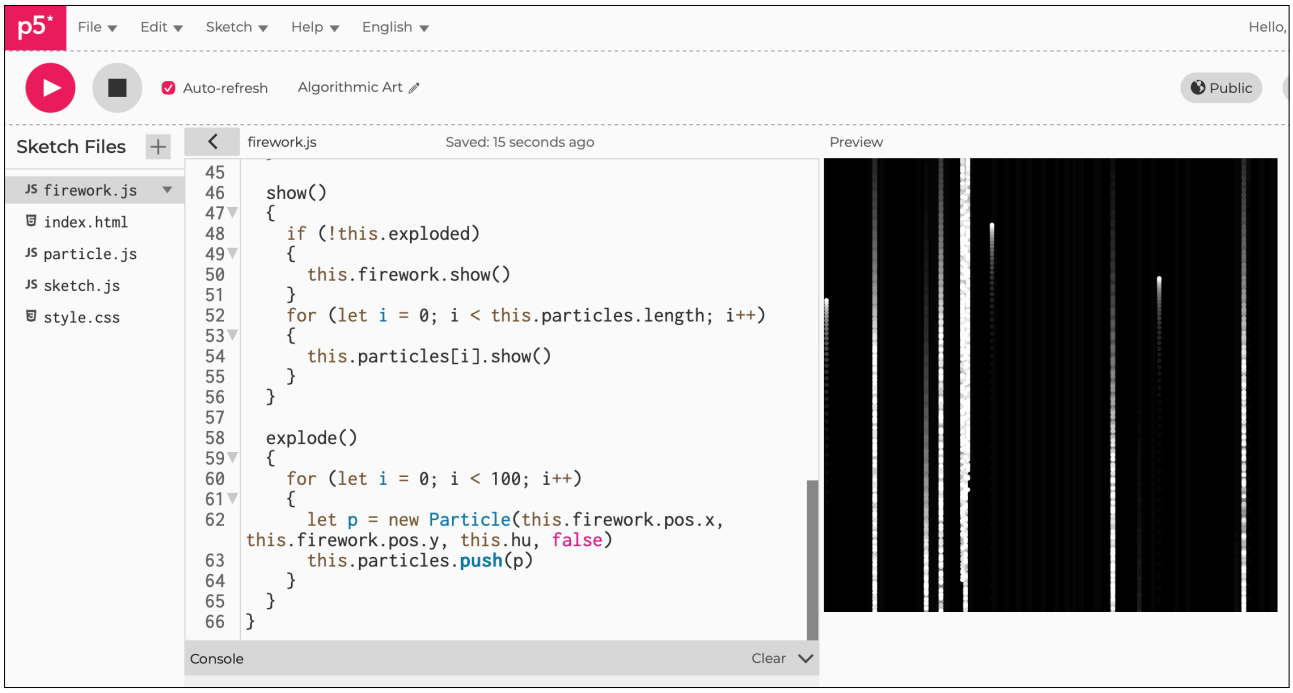
```
class Firework
{
  constructor()
  {
    this.hu = random(255)
    this.firework = new Particle(random(width), height, this.hu, true)
    this.exploded = false
    this.particles = []
  }

  done()
  {
    if (this.exploded && this.particles.length === 0)
    {
      return true
    }
    else
    {
      return false
    }
  }

  move()
  {
    if (!this.exploded)
    {
      this.firework.applyForce(gravity)
      this.firework.move()
      if (this.firework.vel.y >= 0)
      {
```

```
        this.exploded = true
        this.explode()
      }
    }
    for (let i = this.particles.length - 1; i >= 0; i--)
    {
      this.particles[i].applyForce(gravity)
      this.particles[i].move()
      if (this.particles[i].done())
      {
        this.particles.splice(i, 1)
      }
    }
  }

  show()
  {
    if (!this.exploded)
    {
      this.firework.show()
    }
    for (let i = 0; i < this.particles.length; i++)
    {
      this.particles[i].show()
    }
  }

  explode()
  {
    for (let i = 0; i < 100; i++)
    {
      let p = new Particle(this.firework.pos.x, this.firework.pos.y, this.hu,
false)
      this.particles.push(p)
    }
  }
}
```

Notes

This will break it. Don't panic, we will resolve it shortly.

Figure E5.38



Sketch E5.39 adding the colour in

❗ particle.js
In the particle.js sketch, we add it in this way.

particle.js

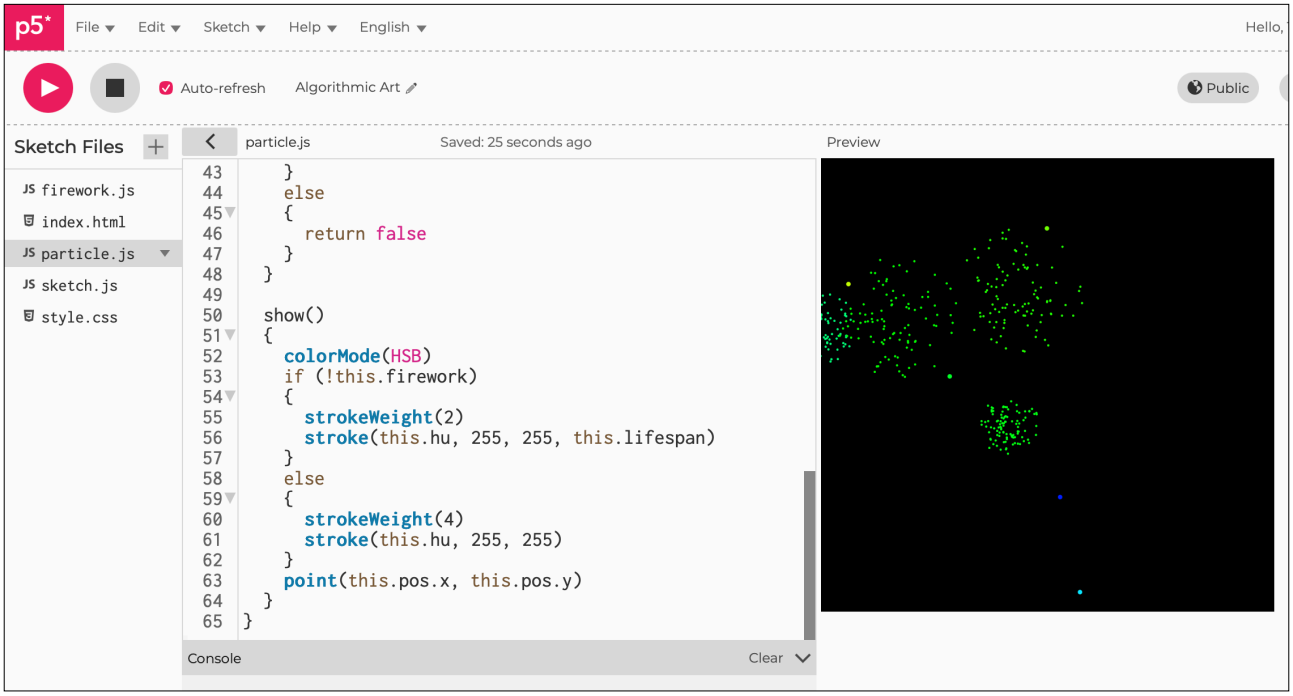
```
class Particle
{
  constructor(x, y, hu, firework)
  {
    this.pos = createVector(x, y)
    this.firework = firework
    this.lifespan = 255
    this.hu = hu
    if (this.firework)
    {
      this.vel = createVector(0, random(-12, -8))
    }
    else
    {
      this.vel = p5.Vector.random2D()
      this.vel.mult(random(2, 10))
    }
    this.acc = createVector(0, 0)
  }

  applyForce(force)
  {
    this.acc.add(force)
  }

  move()
  {
    if (!this.firework)
    {
      this.vel.mult(0.85)
      this.lifespan -= 4
    }
```

```
    this.vel.add(this.acc)
    this.pos.add(this.vel)
    this.acc.mult(0)
  }

  done()
  {
    if (this.lifespan < 0)
    {
      return true
    }
    else
    {
      return false
    }
  }

  show()
  {
    colorMode(HSB)
    if (!this.firework)
    {
      strokeWeight(2)
      stroke(this.hu, 255, 255, this.lifespan)
    }
    else
    {
      strokeWeight(4)
      stroke(this.hu, 255, 255)
    }
    point(this.pos.x, this.pos.y)
  }
}
```

🗒️ Notes

And we are back again, this time in colour.

Figure E5.39


p5*
File
Edit
Sketch
Help
English
Hello,
Auto-refresh
Algorithmic Art
Public
Sketch Files
firework.js
index.html
particle.js
sketch.js
style.css
particle.js
Saved: 25 seconds ago
Preview
43 }
44 else
45 {
46 return false
47 }
48 }
49
50 show()
51 {
52 colorMode(HSB)
53 if (!this.firework)
54 {
55 strokeWeight(2)
56 stroke(this.hu, 255, 255, this.lifespan)
57 }
58 else
59 {
60 strokeWeight(4)
61 stroke(this.hu, 255, 255)
62 }
63 point(this.pos.x, this.pos.y)
64 }
65 }
Console
Clear

Sketch E5.40 colorMode(RGB)

❗ sketch.js

But we have lost the trails, so we need to change the `colorMode()` to `RGB` in sketch.js

sketch.js

```
let fireworks = []
let gravity

function setup()
{
  createCanvas(400, 400)
  gravity = createVector(0, 0.2)
  stroke(255)
  strokeWeight(4)
  background(0)
}

function draw()
{
  colorMode(RGB)
  background(0, 25)
  if (random(1) < 0.05)
  {
    fireworks.push(new Firework())
  }
  for (let i = fireworks.length - 1; i >= 0; i--)
  {
    fireworks[i].move()
    fireworks[i].show()
    if (fireworks[i].done())
    {
      fireworks.splice(i, 1)
    }
  }
}
```

Notes

Now that looks even better. We are done. You have your fireworks display.

Figure E5.40