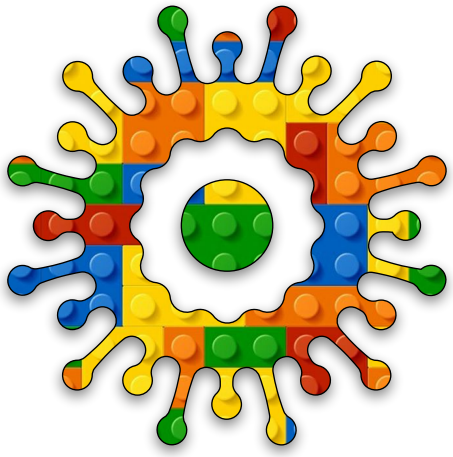
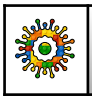
Physics Simulation Module B Unit #2 pendulums



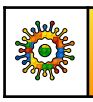


Content

Module A Unit #2 pendulum

Introduction to the pendulum

Making a simple pendulum, which might sound simple enough, but has many factors to consider. An exercise that uses the code you have learnt and then applies it to a simple simulation. This is from Coding Challenge #159 simple pendulum (coding train YouTube channel).



Sketch A2.1 the arm and the bob

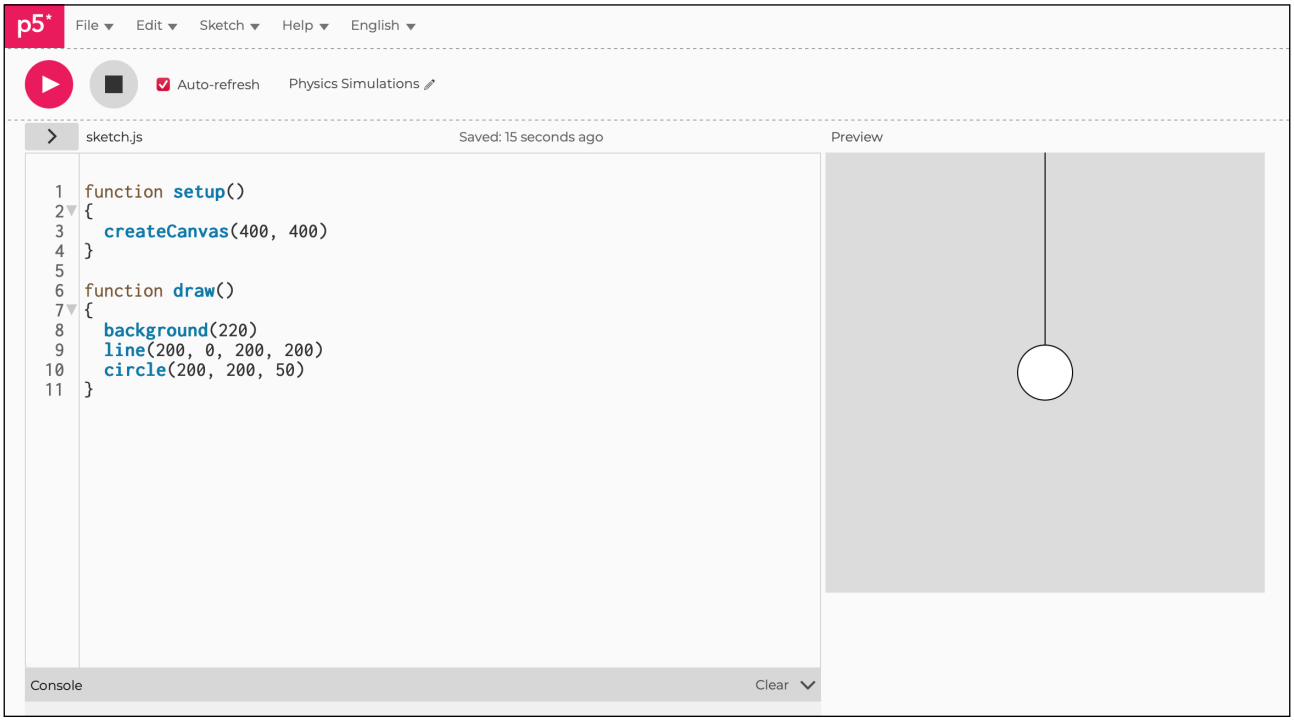
Creating an arm (line) and a bob (circle), very simple.

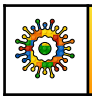
```
function setup()
{
  createCanvas(400, 400)
}

function draw()
{
  background(220)
  line(200, 0, 200, 200)
  circle(200, 200, 50)
}
```

Notes

A static pendulum.

Figure A2.1


p5*
File
Edit
Sketch
Help
English
Auto-refresh
Physics Simulations
sketch.js
Saved: 15 seconds ago
Preview
function setup()
{
createCanvas(400, 400)
}
function draw()
{
background(220)
line(200, 0, 200, 200)
circle(200, 200, 50)
}
Console
Clear

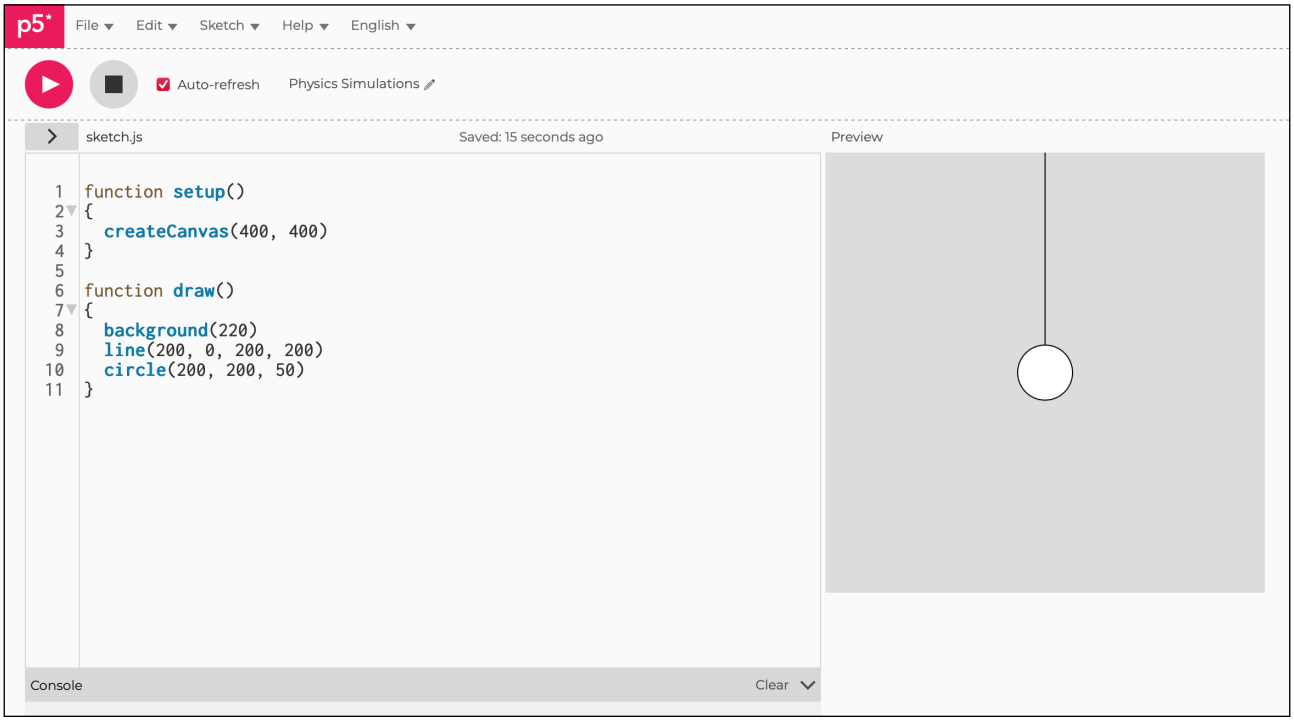


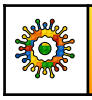
Sketch A2.2 adding variables

The angle to the vertical is PI/4 (π/4), the length (len) is 200, the bob is a vector and the origin is a vector at (200, 0).

```
let angle
let bob
let len
let origin

function setup()
{
  createCanvas(400, 400)
  angle = PI/4
  bob = createVector()
  len = 200
  origin = createVector(200, 0)
}

function draw()
{
  background(220)
  line(200, 0, 200, 200)
  circle(200, 200, 50)
}
```

Figure A2.2


p5*
File
Edit
Sketch
Help
English
Auto-refresh
Physics Simulations
sketch.js
Saved: 15 seconds ago
Preview
1 function setup()
2 {
3 createCanvas(400, 400)
4 }
5
6 function draw()
7 {
8 background(220)
9 line(200, 0, 200, 200)
10 circle(200, 200, 50)
11 }
Console
Clear

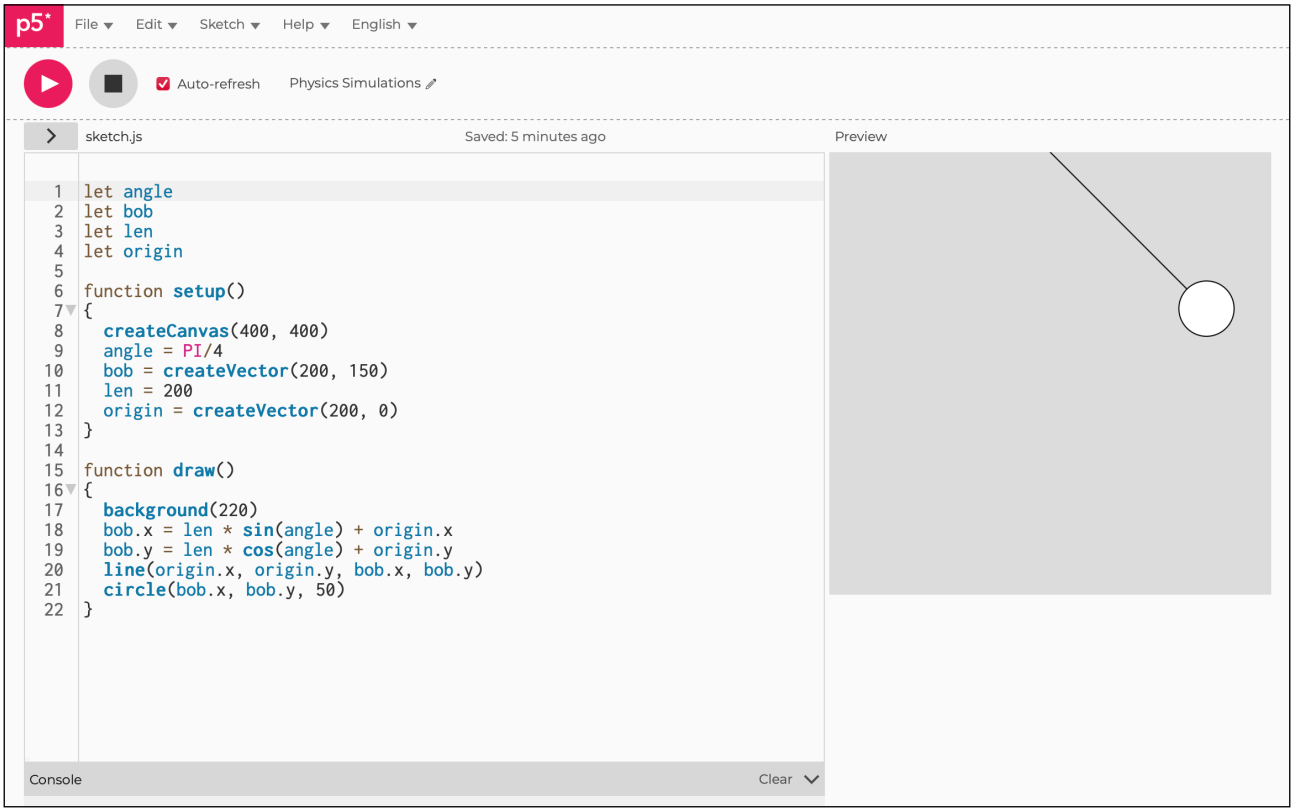


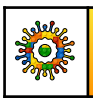
Sketch A2.3 using sine and cosine

Now to add in the calculations for `bob.x` and `bob.y` using `sin(angle)` and `cos(angle)`. We add the `origin.x` and `origin.y` because the angle is measured from `(300, 200)` and not `(0, 0)`.

```
let angle
let bob
let len
let origin

function setup()
{
  createCanvas(400, 400)
  angle = PI/4
  bob = createVector(200, 150)
  len = 200
  origin = createVector(200, 0)
}

function draw()
{
  background(220)
  bob.x = len * sin(angle) + origin.x
  bob.y = len * cos(angle) + origin.y
  line(origin.x, origin.y, bob.x, bob.y)
  circle(bob.x, bob.y, 50)
}
```

Figure A2.3


p5*
File
Edit
Sketch
Help
English
Auto-refresh
Physics Simulations
sketch.js
Saved: 5 minutes ago
Preview
let angle
let bob
let len
let origin
function setup()
{
createCanvas(400, 400)
angle = PI/4
bob = createVector(200, 150)
len = 200
origin = createVector(200, 0)
}
function draw()
{
background(220)
bob.x = len * sin(angle) + origin.x
bob.y = len * cos(angle) + origin.y
line(origin.x, origin.y, bob.x, bob.y)
circle(bob.x, bob.y, 50)
}
Console
Clear



Sketch A2.4 angular velocity & acceleration

Let's add angular velocity (angleV) and acceleration (angleA).

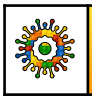
```
let angle
let bob
let len
let origin
let angleV = 0
let angleA = 0

function setup()
{
  createCanvas(400, 400)
  angle = PI/4
  bob = createVector(200, 150)
  len = 200
  origin = createVector(200, 0)
}

function draw()
{
  background(220)
  angleV += angleA
  angle += angleV
  bob.x = len * sin(angle) + origin.x
  bob.y = len * cos(angle) + origin.y
  line(origin.x, origin.y, bob.x, bob.y)
  circle(bob.x, bob.y, 50)
}
```

Notes

Same as before. Now to add some forces...

🌻 Challenges

1. Give a small value to `angleA`.
2. Give a small value to `angleV`.



Sketch A2.5 gravity

Now add `gravity` to make the simulation realistic. The mass will be `1`, also we create our own value for `gravity` ignoring Newton! This is getting there but still needs more maths applied to this in the next sketch after this one.

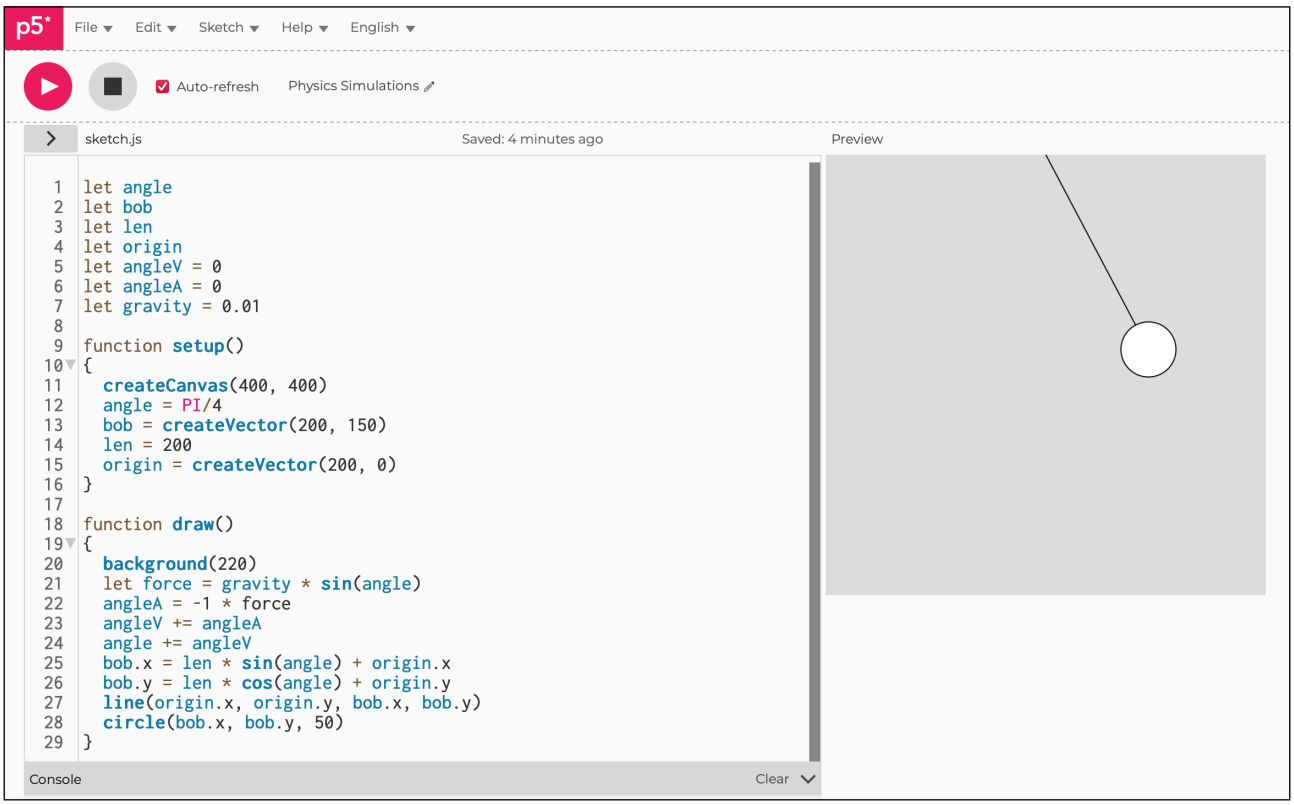
```
let angle
let bob
let len
let origin
let angleV = 0
let angleA = 0
let gravity = 0.01

function setup()
{
  createCanvas(400, 400)
  angle = PI/4
  bob = createVector(200, 150)
  len = 200
  origin = createVector(200, 0)
}

function draw()
{
  background(220)
  let force = gravity * sin(angle)
  angleA = -1 * force
  angleV += angleA
  angle += angleV
  bob.x = len * sin(angle) + origin.x
  bob.y = len * cos(angle) + origin.y
```

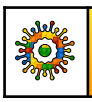
```
  line(origin.x, origin.y, bob.x, bob.y)
  circle(bob.x, bob.y, 50)
}
```

🗒️ Notes

You should now see it swinging backwards and forwards, but not very realistically. There is no loss of energy.

Figure A2.5


p5*
File
Edit
Sketch
Help
English
Auto-refresh
Physics Simulations
sketch.js
Saved: 4 minutes ago
Preview
let angle
let bob
let len
let origin
let angleV = 0
let angleA = 0
let gravity = 0.01
function setup()
{
createCanvas(400, 400)
angle = PI/4
bob = createVector(200, 150)
len = 200
origin = createVector(200, 0)
}
function draw()
{
background(220)
let force = gravity * sin(angle)
angleA = -1 * force
angleV += angleA
angle += angleV
bob.x = len * sin(angle) + origin.x
bob.y = len * cos(angle) + origin.y
line(origin.x, origin.y, bob.x, bob.y)
circle(bob.x, bob.y, 50)
}
Console
Clear



Sketch A2.6 realism

We improve the angular acceleration to be more realistic. We add `gravity` and give it a value of `1` (this is just for starters - you can play around with that value). The angle acceleration is divided by its length (`len`). Also adding some dampening with:
`angleV *= 0.995`

This helps it to decay by a small amount each swing.

```
let angle
let bob
let len
let origin
let angleV = 0
let angleA = 0
let gravity = 1

function setup()
{
  createCanvas(400, 400)
  angle = PI/4
  bob = createVector(200, 150)
  len = 200
  origin = createVector(200, 0)
}

function draw()
{
  background(220)
  let force = gravity * sin(angle)
  angleA = (-1 * force) / len
  angleV += angleA
```

```
  angle += angleV
  angleV *= 0.995
  bob.x = len * sin(angle) + origin.x
  bob.y = len * cos(angle) + origin.y
  line(origin.x, origin.y, bob.x, bob.y)
  circle(bob.x, bob.y, 50)
}
```

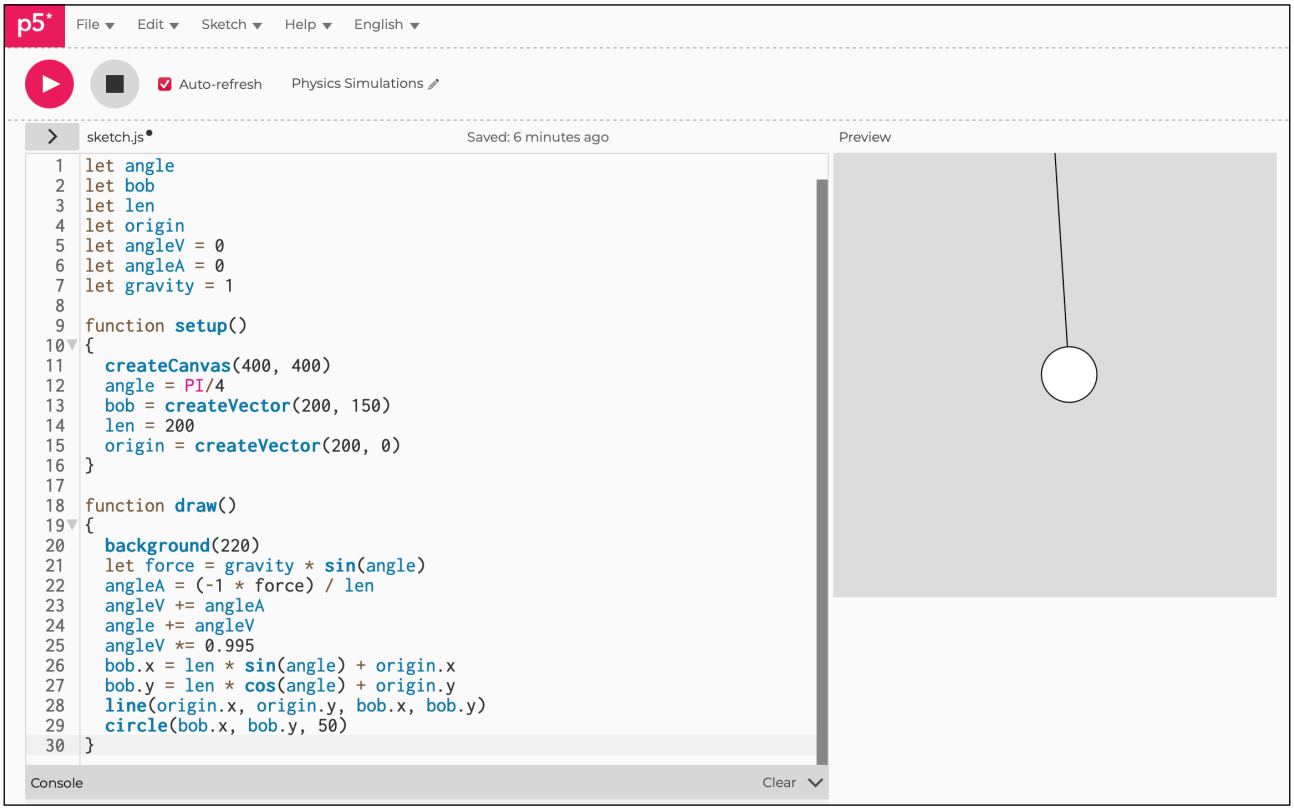
🗒️ Notes

So just when you thought that making a simple pendulum would be simple... This is a simulation and has to take into account the necessary forces to be anywhere near a physics engine; by that, I mean it simulates physics in the real world.

🌻 Challenges

1. Change the amount of dampening/decay or comment it out completely.
2. Try different lengths of string - is it still realistic?

Figure A2.6


p5*
File
Edit
Sketch
Help
English
Auto-refresh
Physics Simulations
sketch.js
Saved: 6 minutes ago
Preview
let angle
let bob
let len
let origin
let angleV = 0
let angleA = 0
let gravity = 1
function setup()
{
createCanvas(400, 400)
angle = PI/4
bob = createVector(200, 150)
len = 200
origin = createVector(200, 0)
}
function draw()
{
background(220)
let force = gravity * sin(angle)
angleA = (-1 * force) / len
angleV += angleA
angle += angleV
angleV *= 0.995
bob.x = len * sin(angle) + origin.x
bob.y = len * cos(angle) + origin.y
line(origin.x, origin.y, bob.x, bob.y)
circle(bob.x, bob.y, 50)
}
Console
Clear