Luxembourg Tech School

LEVEL GO
2025-2026

4 – Mini-Game with p5.js

innovative initiatives digital luxembourg. · SCRIPT · cgie centre de gestion informatique de l'éducation · LSA Luxembourg Space Agency · Luxembourg National Research Fund · André Losch Fondation · Fondation BBL pour l'éducation financière · State Street · funded by œuvre nationale · Luxair · the LHoFT · Luxembourg-City Incubator

Recap Quiz

1. What is the purpose of the *draw()* function in general?

mySketch

```
1   function setup() {
2     createCanvas(600, 600);
3     background(180);
4   }
5
6   function draw() {
7     circle(100, 80, 80);
8   }
```

A. It runs once

B. It draws the background

C. It makes shapes bounce

D. It runs 60× per second

1. What is the purpose of the *draw()* function in general?

mySketch

```
1  function setup() {
2    createCanvas(600, 600);
3    background(180);
4  }
5
6  function draw() {
7    circle(100, 80, 80);
8  }
```

A. It runs once

B. It draws the background

C. It makes shapes bounce

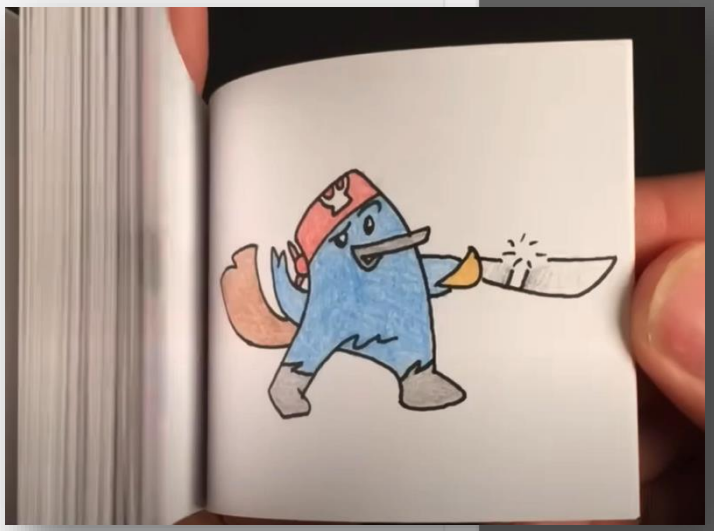
D. It runs 60× per second

The Magic of draw()

happens only 1x

happens 60x per second

```
let x = 100;

function setup() {
  createCanvas(windowWidth, windowHeight);
}

function draw() {
  background(100);
  fill(250, 250, 0);
  circle(x, 200, 50);
  x = x + 2;
}
```



2. What happens here?

```
x = x + 2;
```

A. x resets to 2

B. x gets bigger by 2 every frame

C. x stops changing

D. x stays the same

2. What happens here?

```
x = x + 2;
```

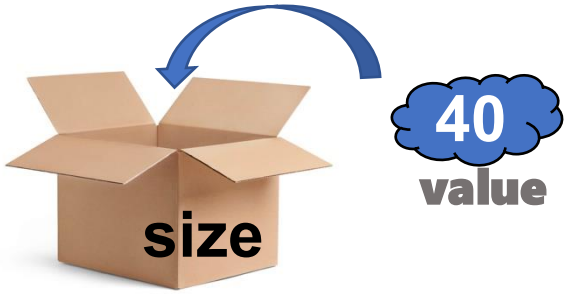
A. x resets to 2

B. x gets bigger by 2 every frame

C. x stops changing

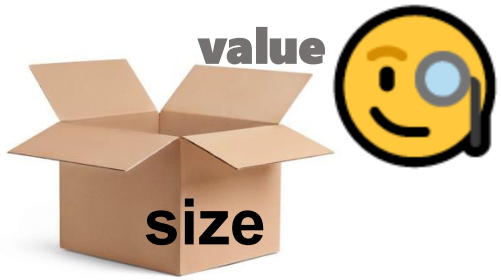
D. x stays the same

3 ways of using variables



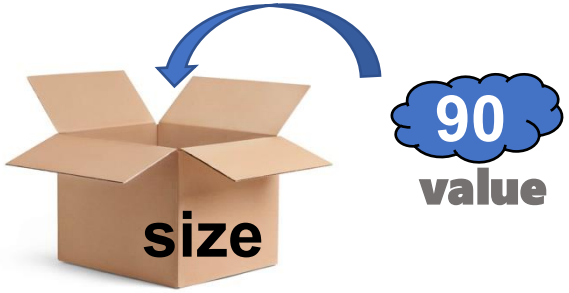
Create define name and value at the top of your code

```
let size = 40;
```



Read Use the variable to put its value into your code

```
circle(100, 80, size);
```



Change Put a new value into the variable

```
size = 90;
```

3. What does this condition check?

```
if (x > width)
```

A. Is x moving?

B. Is x off the screen on the right?

C. Is x too small?

D. Is x equal to width?

3. What does this condition check?

```
if (x > width)
```

A. Is x moving?

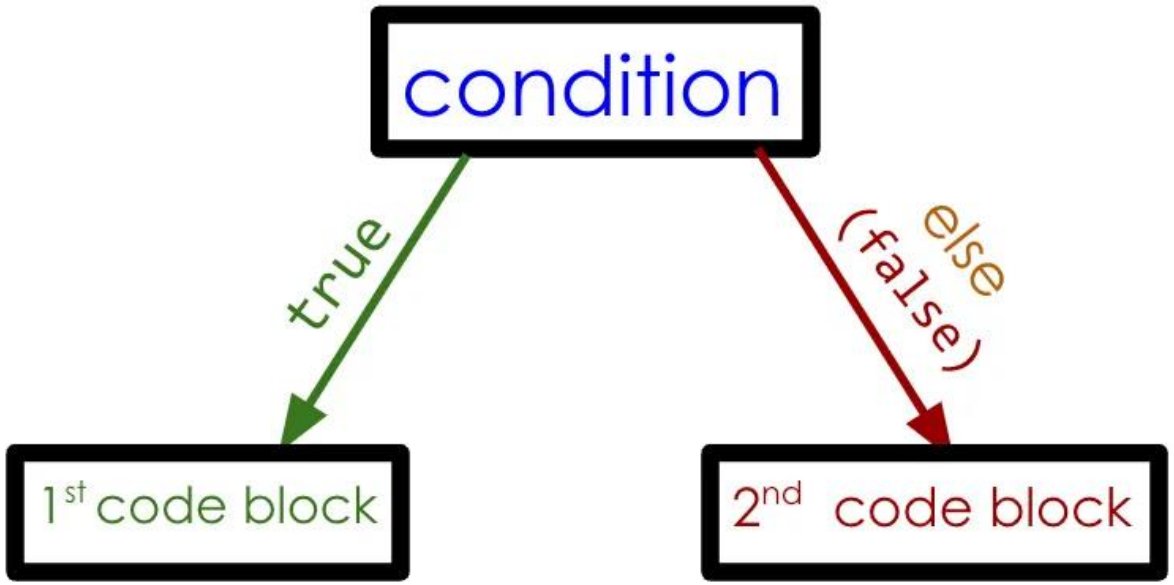
B. Is x off the screen on the right?

C. Is x too small?

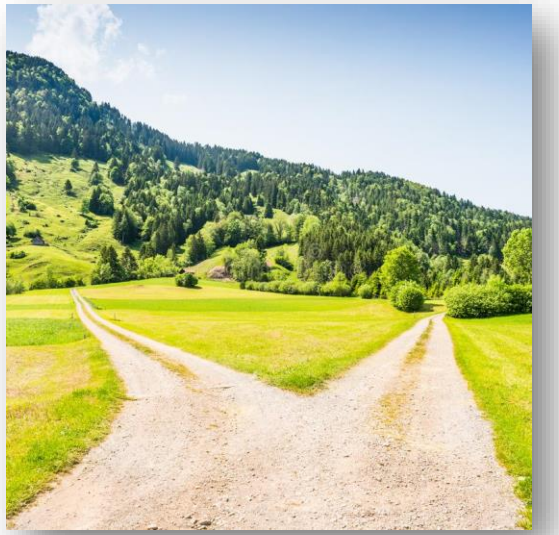
D. Is x equal to width?

Different paths with *if...else...*

How our programs can follow different paths.




```
if (condition) {
  // code to run if the condition is true
} else {
  // code to run if the condition is false
}
```



4. What does this do?

```
fill(random(255), random(255), random(255));
```

A. Deletes the shape

B. Moves the shape

C. Sets a random colour

D. Sets a black fill

4. What does this do?

```
fill(random(255), random(255), random(255));
```

A Deletes the shape

B Moves the shape

C Sets a random colour

D Sets a black fill

random(*min, max*)

We will need the ***random( )*** function to generate random values.

Example

```
random(0,255);
```

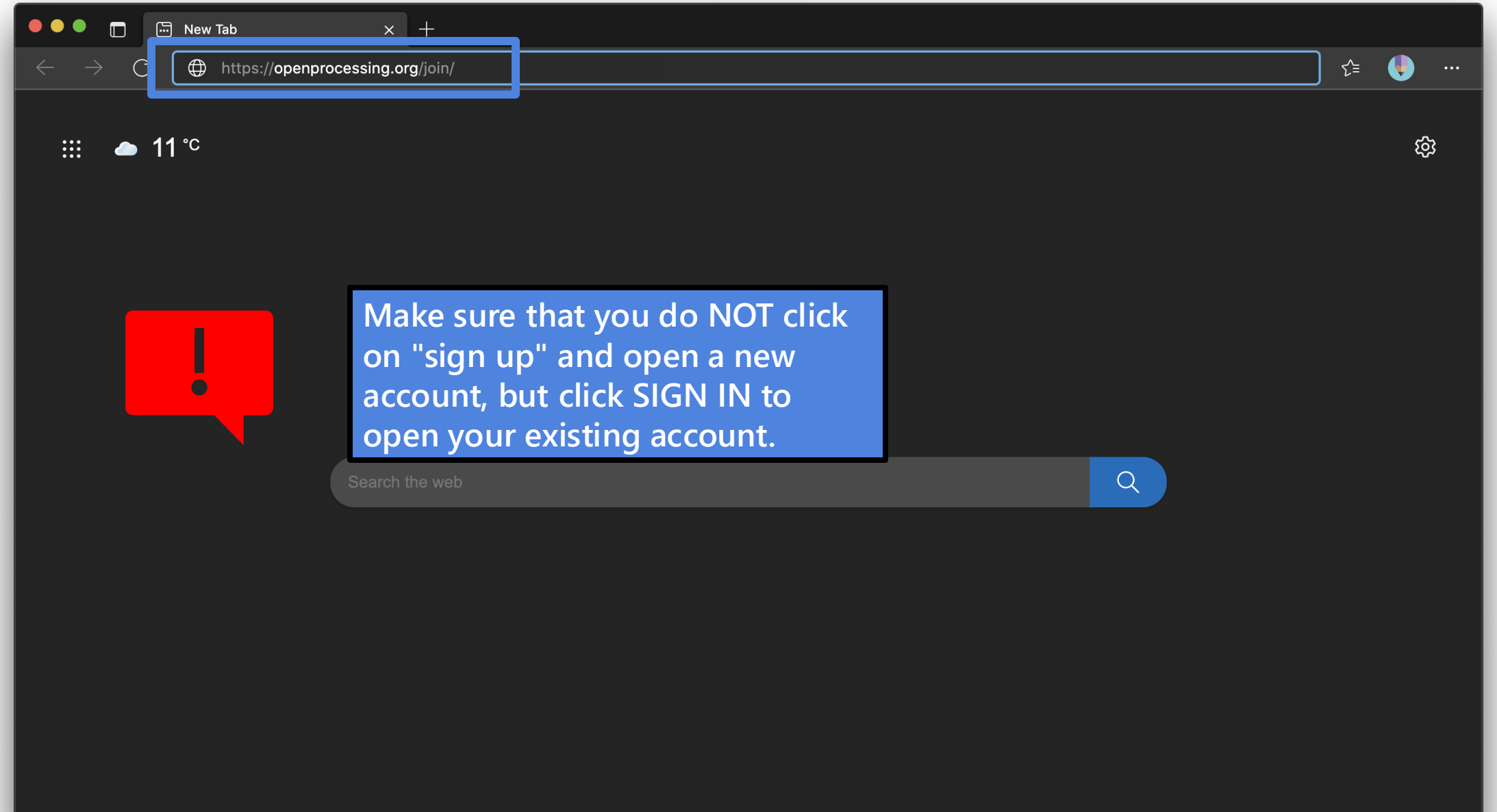
generates a random value between 0 and 254

Let's code our first game

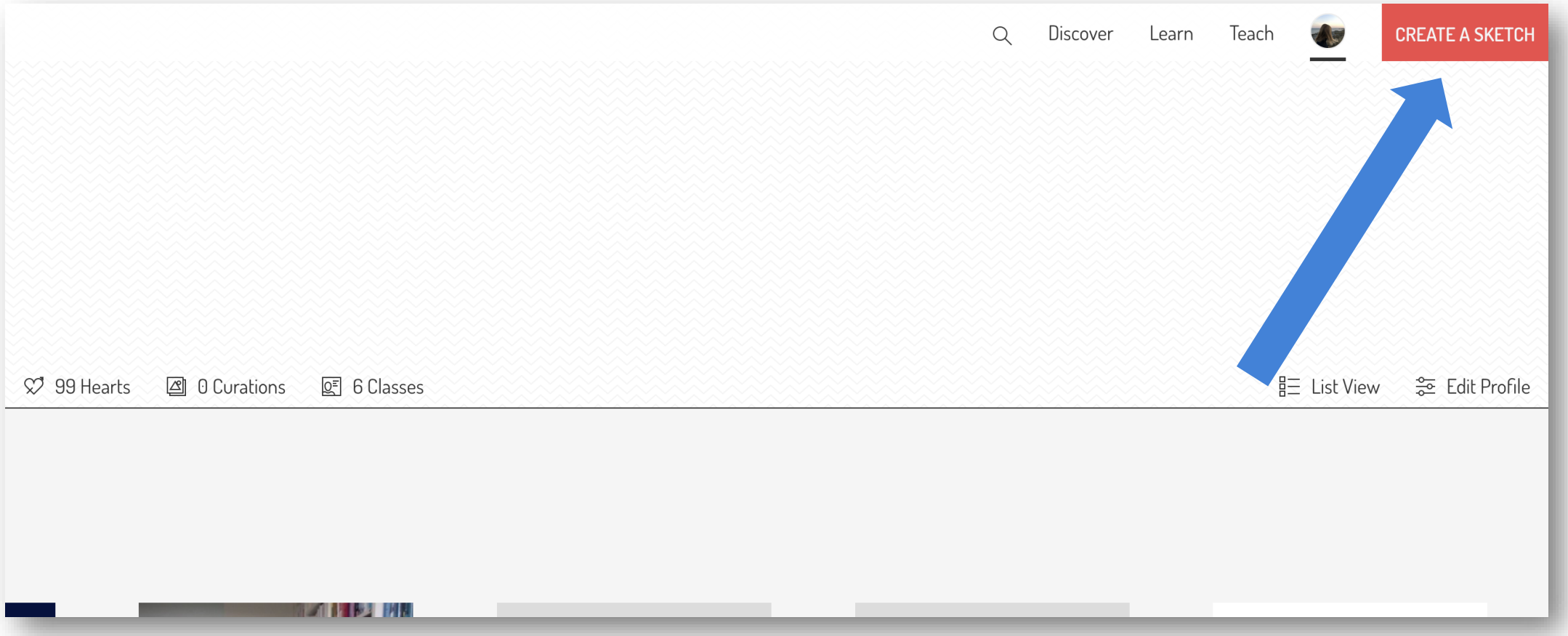
Code a Clicker-Game

You program a simple but cool game where you click on an ever-moving, color-changing circle to earn points.

Open OpenProcessing in your browser



Create a new sketch



LIVE CODING

1: Draw a circle using variables

```
let x = 100;
let y = 300;

function setup() {
  createCanvas(600, 600);
  noStroke();
}

function draw() {
  background(10, 60, 150);
  fill(250, 220, 130);
  circle(x, y, 80);
}
```

Recap

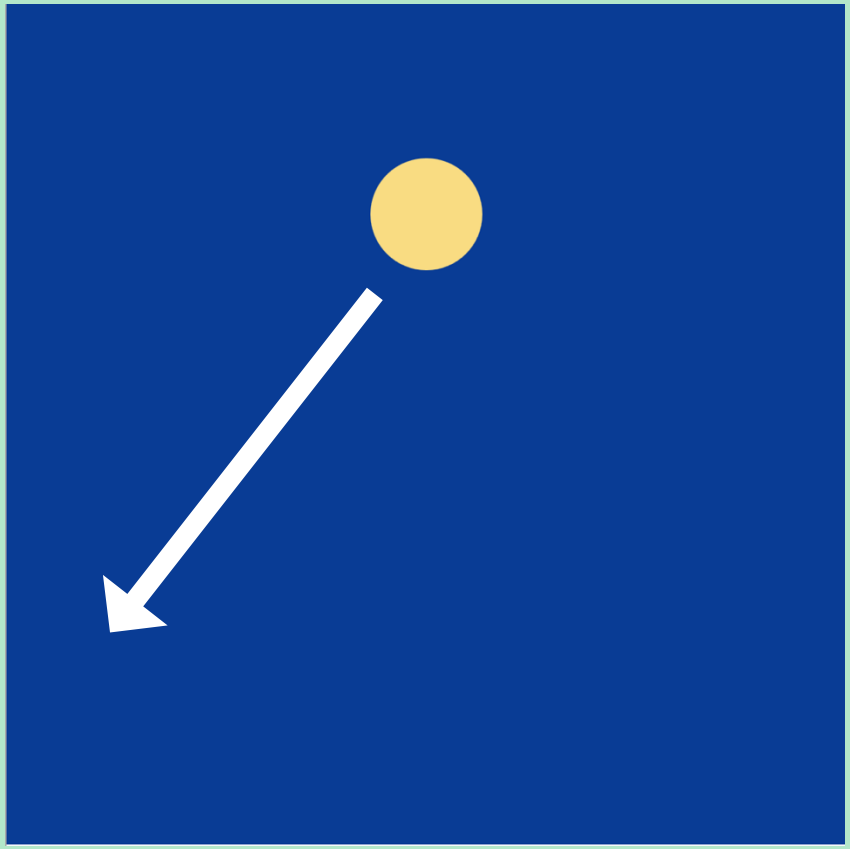
- initiate two variables *x* and *y*
- *x* and *y* control the position of the circle
- *circle(x, y, size)* draws a circle at those coordinates
- *draw()* runs in a loop → it redraws every frame; 60x per second

Design choices

- Choose a *background()* colour
- Choose a *fill()* colour for the circle
- *noStroke()* deletes the stroke around the circle

Remember how we move a circle?

What do we need to move the circle?

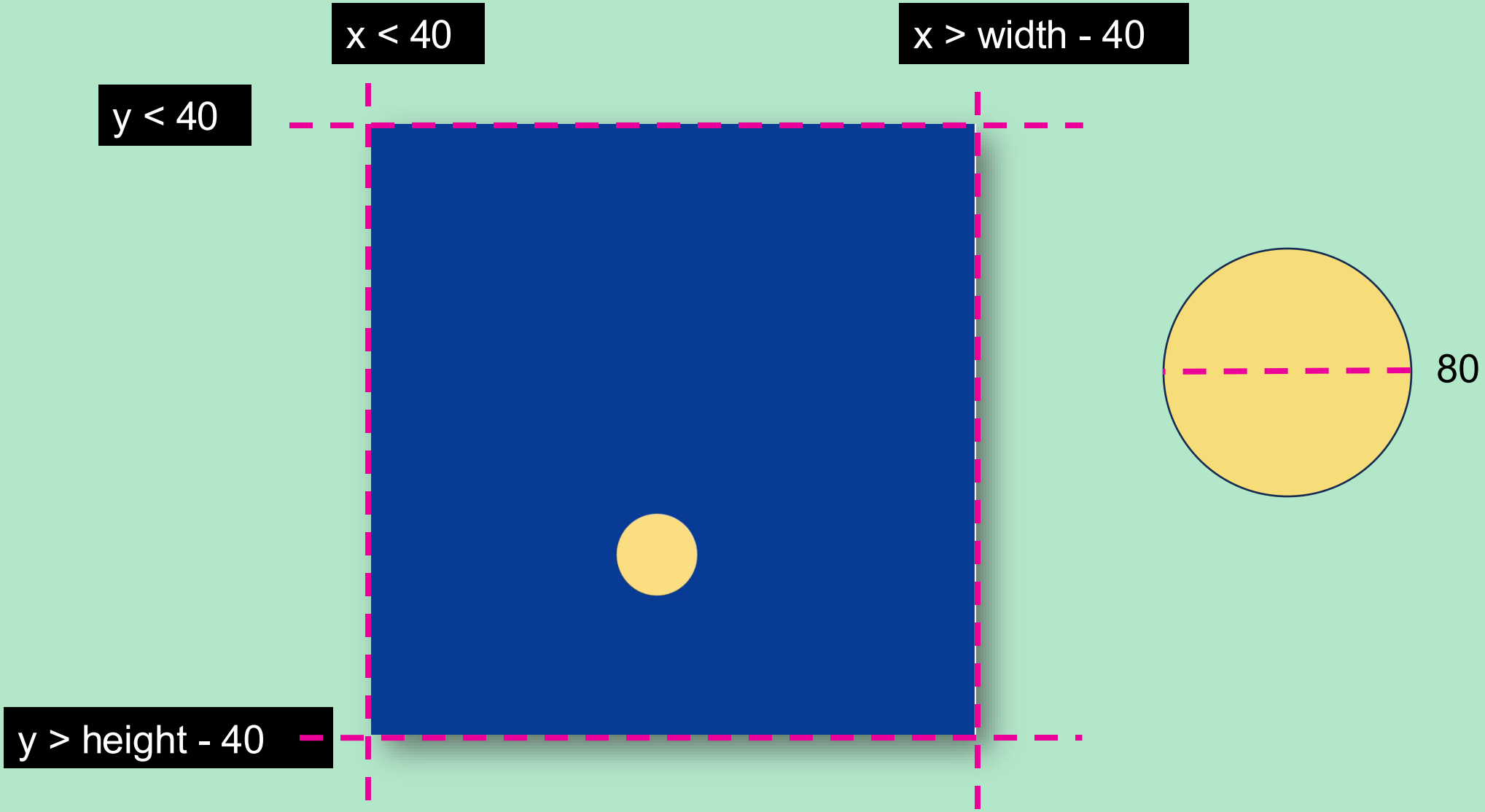


Think about what values need to be changed?

2: Make the circle move

```
let x = 100;
let y = 300;

function setup() {
  createCanvas(600, 600);
  noStroke();
}

function draw() {
  background(10, 60, 150);
  fill(250, 220, 130);
  circle(x, y, 80);
  x = x + 3;
  y = y + 2;
}
```

Remember how to bounce off the edges?



3: Bounce off the canvas edges

```
let x = 100;
let y = 300;
let xSpeed = 3;
let ySpeed = 2;

function setup() {
  createCanvas(600, 600);
  noStroke();
}

function draw() {
  background(10, 60, 150);
  fill(250, 220, 130);
  circle(x, y, 80);
  x += xSpeed;
  y += ySpeed;

  if (x > width - 40 || x < 40) {
    xSpeed = -xSpeed;
  }

  if (y > height - 40 || y < 40) {
    ySpeed = -ySpeed;
  }
}
```

We add two speed variables.

Hint: *x += xSpeed;* is the same as *x = x + xSpeed;*

We add an *if* statement to check if our *x* value is greater than the (width – 40) OR less than 40.

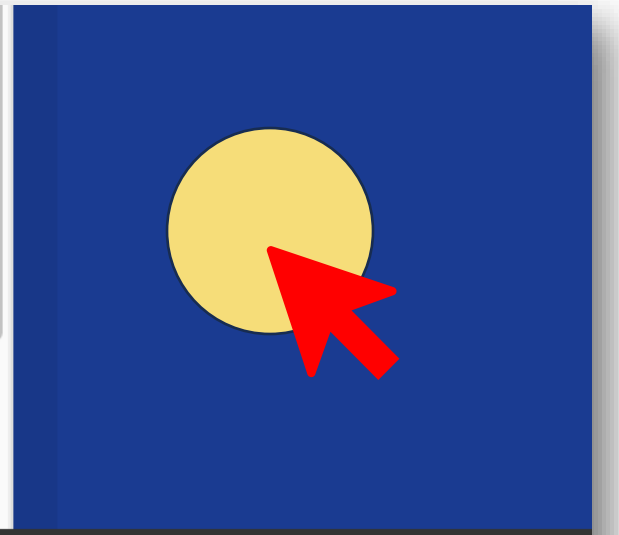
-xSpeed flips direction = bounce!

We add a 2nd *if* statement to check the same for y.

Intro to *mousePressed()*

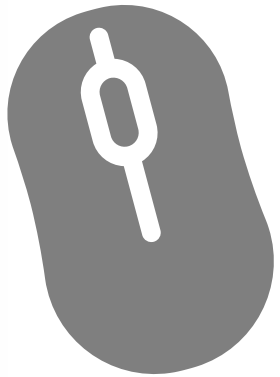
We want to **click on the moving shape** to score points!

```
      ySpeed = -ySpeed;
    }
  }

  function mousePressed() {
    print("You clicked!");
  }

```



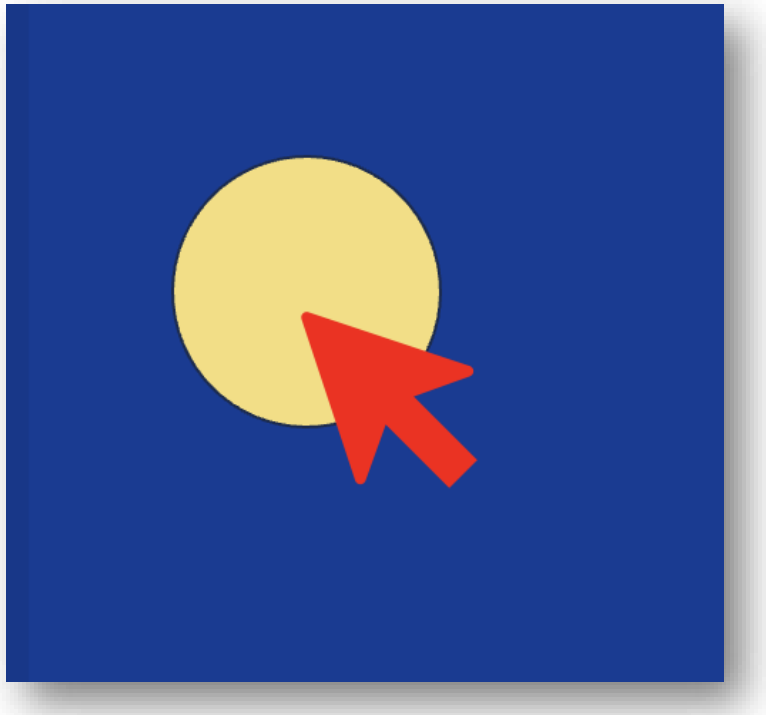
```
You clicked!
```

- *mousePressed()* runs when the mouse is clicked
- We can use *mouseX* and *mouseY* to check where it was clicked



Detecting hits with *dist()*

To know if we clicked **on the circle**, we need to check how close the mouse is to the circle's center.



dist() calculates the distance between two points.

```
dist(mouseX, mouseY, x, y);
```

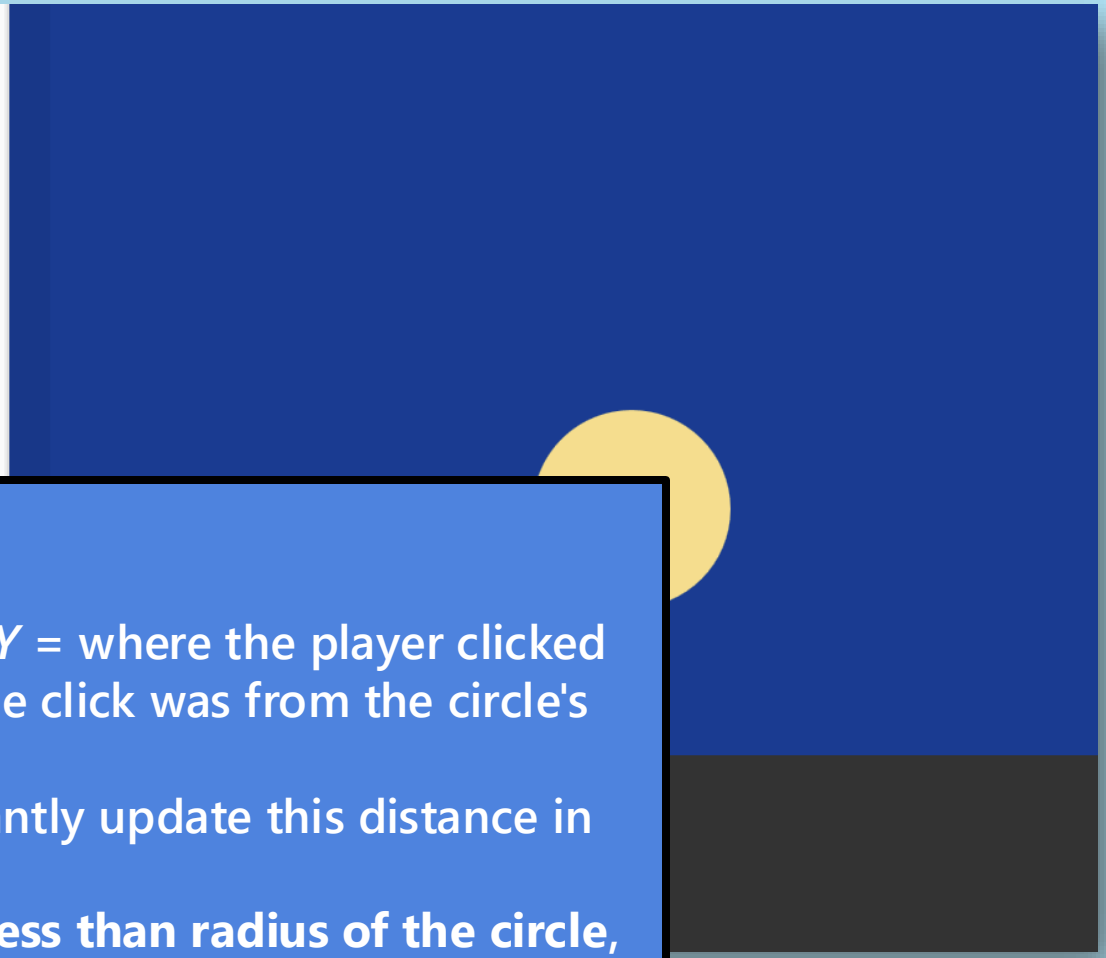
position of our mouse

position of the circle

4: Click detection with *mousePressed()*

```
23        ySpeed = -ySpeed;
24      }
25    }
26
27    function mousePressed() {
28      let d = dist(mouseX, mouseY, x, y);
29
30      if (d < 40) {
31        print("Hit!");
32      }
33    }
34
```

```
4x Hit!
```



Explain:

- *mouseX* and *mouseY* = where the player clicked
- *dist(...)* = how far the click was from the circle's center
- We save and constantly update this distance in variable *d*
- If (and only if) *d* is **less than radius of the circle**, it's a hit!

5: Add a score counter

At the top, add a ***score*** variable:

mySketch

```
let x = 100;
let y = 300;
let xSpeed = 3;
let ySpeed = 2;
let score = 0;
```

1

Update inside *mousePressed()* with ***score += 1;*** to increase the score by 1 if you click on the circle.

```
function mousePressed() {
  let d = dist(mouseX, mouseY, x, y);

  if (d < 40) {
    score += 1;
  }
}
```

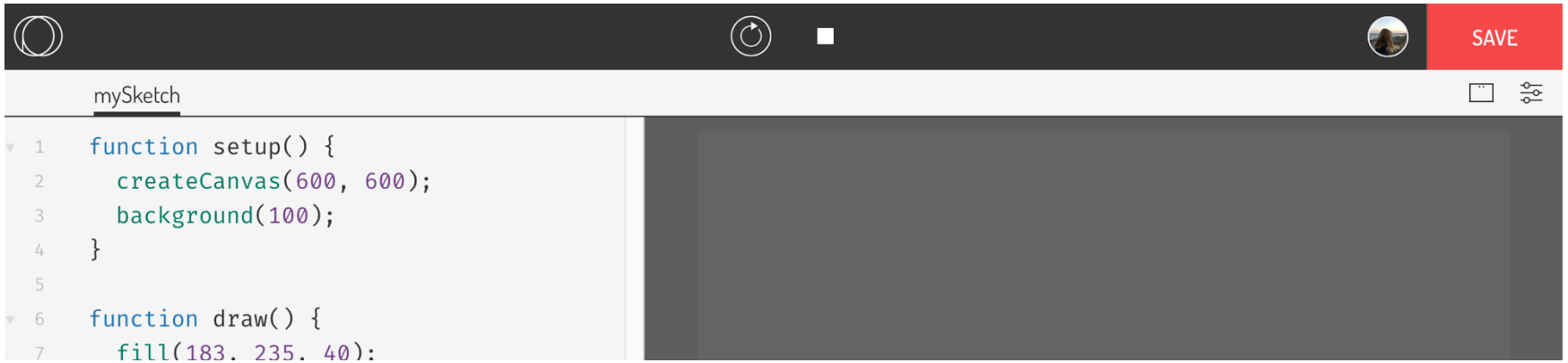
2

Display the current ***score*** inside *draw()* with ***text():***

```
function draw() {
  background(10, 60, 150);
  fill(250, 220, 130);
  circle(x, y, 80);
  x += xSpeed;
  y += ySpeed;

  textSize(24);
  text("Score: " + score, 20, 40);

  if (x > width - 40 || x < 40) {
```

3

Don't forget to save

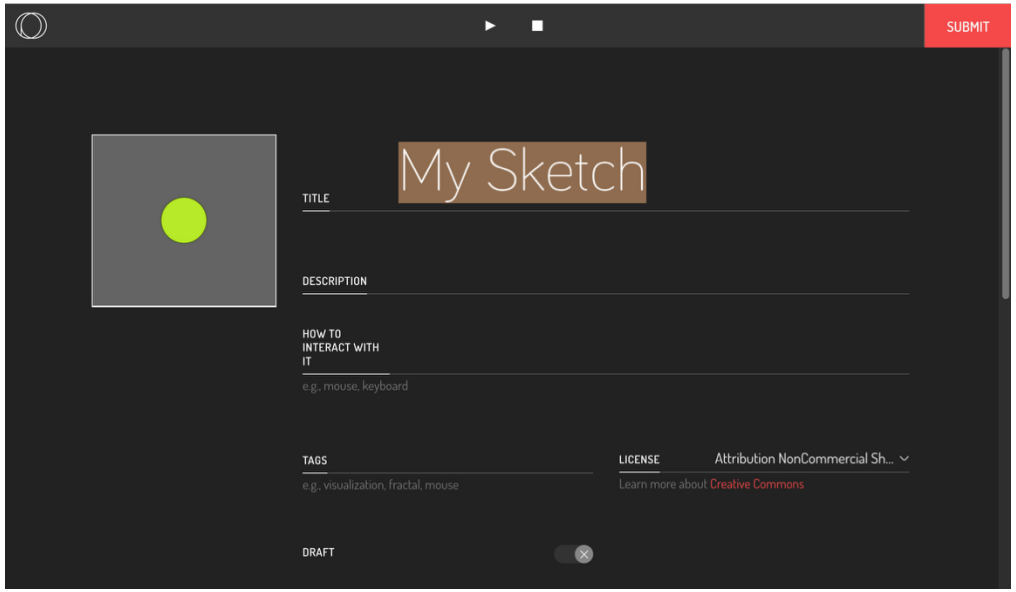


1

Click on SAVE

2

Give the sketch a new title instead of "My Sketch" e.g. "Clicker Game"



3

Click on SUBMIT

Now we have a simple Clicker Game !

How can we make it more exciting?

-
-
-
-
-

Some ideas to improve the game

- Reset circle to a random spot after hit
- Change colour after hit
- Make circle smaller or faster after hit
- Add a “misses” counter if the player clicks and misses
- …..

Try it out and show us your version !

EXTRA: add more difficulty

Hint: reset circle position after hit

```
function mousePressed() {
  let d = dist(mouseX, mouseY, x, y);

  if (d < 40) {
    score += 1;
    x = random(40, width - 40);
    y = random(40, height - 40);
  }
}
```

Hint: change circle colour after hit

Create three variables to store the current RGB values:

Add at the top, so the colour stays consistent until the next hit.

```
let rCol = 250;
let gCol = 220;
let bCol = 130;
```

1

Use these in *fill()* when drawing the circle:

Replace the static ***fill(250, 220, 130);*** with:

```
function draw() {
  background(10, 60, 150);
  fill(rCol, gCol, bCol);
  circle(x, y, 80);
```

2

Change them to random values when the circle is clicked:

Inside your ***mousePressed()*** function, after a successful hit ***(if (d < r))***, add:

```
    x = random(40, width - 40);
    y = random(40, height - 40);

    rCol = random(255);
    gCol = random(255);
    bCol = random(255);
}
```

3

Hint: make circle a bit faster after hit

```
function mousePressed() {
  let d = dist(mouseX, mouseY, x, y);

  if (d < 40) {
    score += 1;
    x = random(40, width - 40);
    y = random(40, height - 40);

    rCol = random(255);
    gCol = random(255);
    bCol = random(255);

    xSpeed *= 1.05;
    ySpeed *= 1.05;
  }
}
```

Hint: shrink circle after a hit

Create a radius variable ***r***

Add at the top and replace every 40 value with r in the code.

```
let y = 300;
let r = 40;
let xSpeed = 3;
```

Attention:

```
circle(x, y, 2*r);
```

1

Inside *mousePressed()* add:

```
  if (d < r) {
    score += 1;
    x = random(r, width - r);
    y = random(r, height - r);

    rCol = random(255);
    gCol = random(255);
    bCol = random(255);

    xSpeed *= 1.05;
    ySpeed *= 1.05;
    r = r * 0.95;
  }
}
```

2

To avoid making the circle **too small**, you can add:

```
    xSpeed *= 1.05;
    ySpeed *= 1.05;
    r = r * 0.95;
    if (r < 10) {
      r = 10;
    }
  }
}
```

3

Hint: add a „Misses“ counter

Create a new variable

```
let score = 0;
let misses = 0;
let rCol = 250;
```

1

Update *mousePressed()*

```
if (d < r) {
  score += 1;
  x = random(r, width - r);
  y = random(r, height - r);

  rCol = random(255);
  gCol = random(255);
  bCol = random(255);

  xSpeed *= 1.05;
  ySpeed *= 1.05;
  r = r * 0.95;
  if (r < 10) {
    r = 10;
  }
} else {
  misses += 1;
}
```

2

Display in *draw()*

```
textSize(24);
text("Score: " + score, 20, 40);
text("Misses: " + misses, 20, 70);
```

3

Final Code

```
let x = 100;
let y = 300;
let r = 40;
let xSpeed = 3;
let ySpeed = 2;
let score = 0;
let misses = 0;
let rCol = 250;
let gCol = 220;
let bCol = 130;

function setup() {
  createCanvas(600, 600);
  noStroke();
}
```

```
function draw() {
  background(10, 60, 150);
  fill(rCol, gCol, bCol);
  circle(x, y, 2*r);
  x += xSpeed;
  y += ySpeed;

  textSize(24);
  text("Score: " + score, 20, 40);
  text("Misses: " + misses, 20, 70);

  if (x > width - r || x < r) {
    xSpeed = -xSpeed;
  }

  if (y > height - r || y < r) {
    ySpeed = -ySpeed;
  }
}
```

```
function mousePressed() {
  let d = dist(mouseX, mouseY, x, y);

  if (d < r) {
    score += 1;
    x = random(r, width - r);
    y = random(r, height - r);

    rCol = random(255);
    gCol = random(255);
    bCol = random(255);

    xSpeed *= 1.05;
    ySpeed *= 1.05;
    r = r * 0.95;
    if (r < 10) {
      r = 10;
    }
  } else {
    misses += 1;
  }
}
```

Luxembourg Tech School