

Uni Studies 3: Intro to Processing 2

Assoc. Professor Donald J. Patterson
Uni Stu 3 Fall 2012



Intro to Processing

<http://processing.org/>

Intro to Processing



<http://processing.org/>

Intro to Processing

- What the heck is Processing?
 - A **programming language**
 - An **environment for running the programs**



<http://processing.org/>

Intro to Processing

- What the heck is Processing?
 - A **programming language**
 - An **environment for running the programs**
- What is it for?
 - It is for people who want to create
 - **images**
 - **animations**
 - **interactions**



<http://processing.org/>

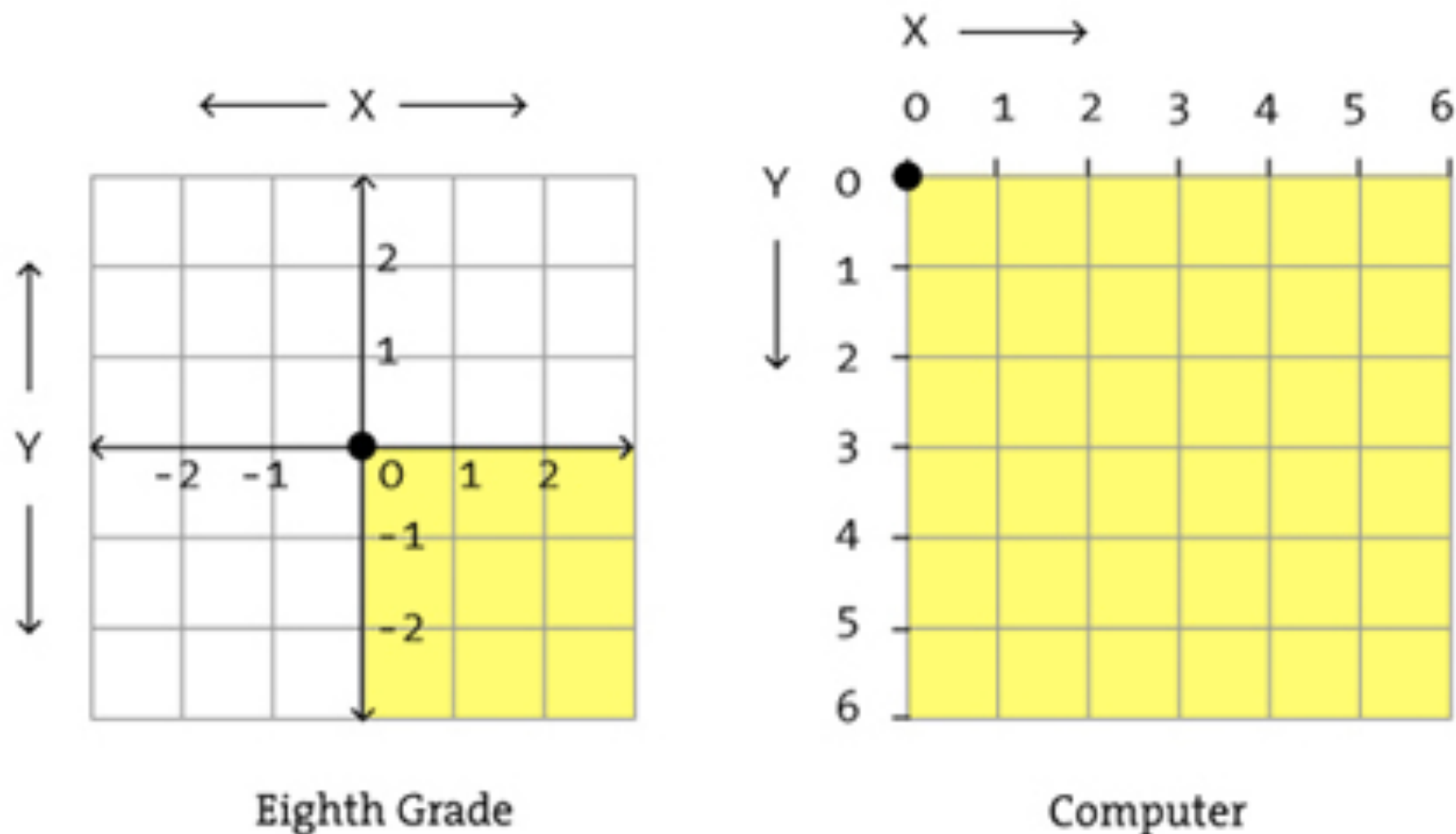
- What the heck is Processing?
 - A **programming language**
 - An **environment for running the programs**
- What is it for?
 - It is for people who want to create
 - **images**
 - **animations**
 - **interactions**
- Who is it for?
 - **students**
 - **artists**
 - **designers**
 - **researchers**
 - **hobbyists**



<http://processing.org/>

Intro to Processing

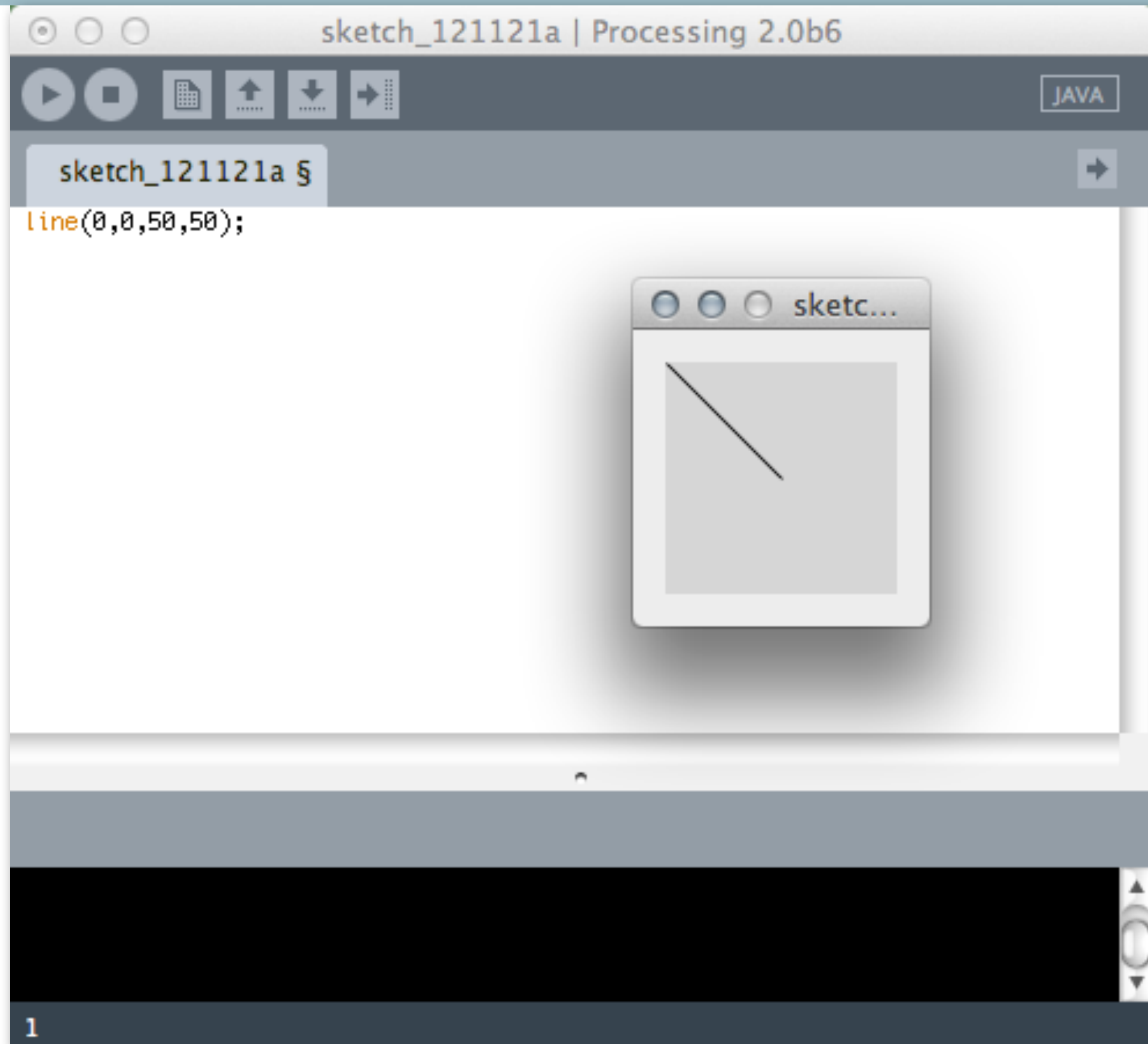
- The computer screen is like a big sheet of graph paper



- The cross-points refer to **pixels**.

<http://processing.org/>

Intro to Processing



<http://processing.org/>

- How do simple shapes get drawn on the computer?

<http://processing.org/>

Intro to Processing

- How do simple shapes get drawn on the computer?



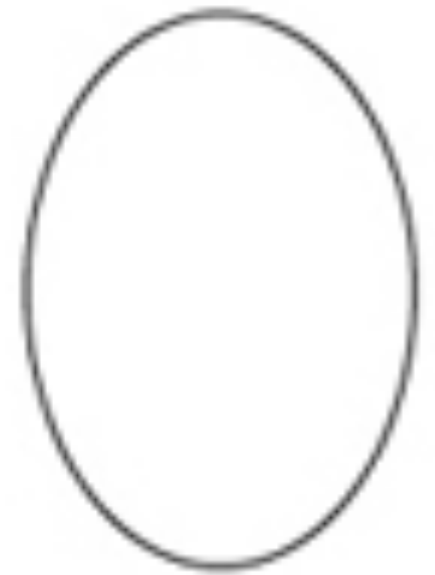
Point



Line



Rectangle



Ellipse

<http://processing.org/>

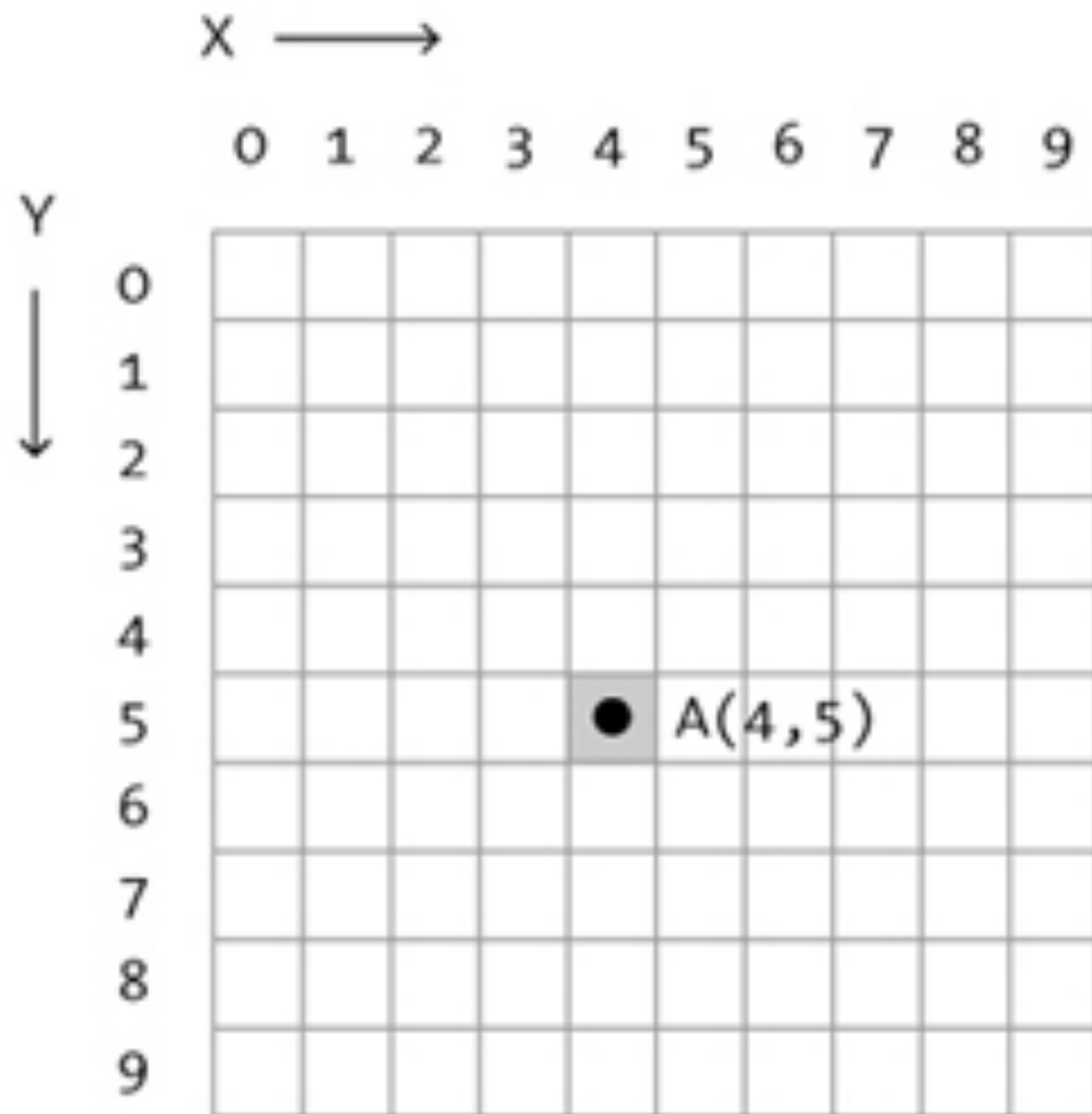
Intro to Processing

- a point
 - point (4,5)

<http://processing.org/>

Intro to Processing

- a point
 - point (4,5)



<http://processing.org/>

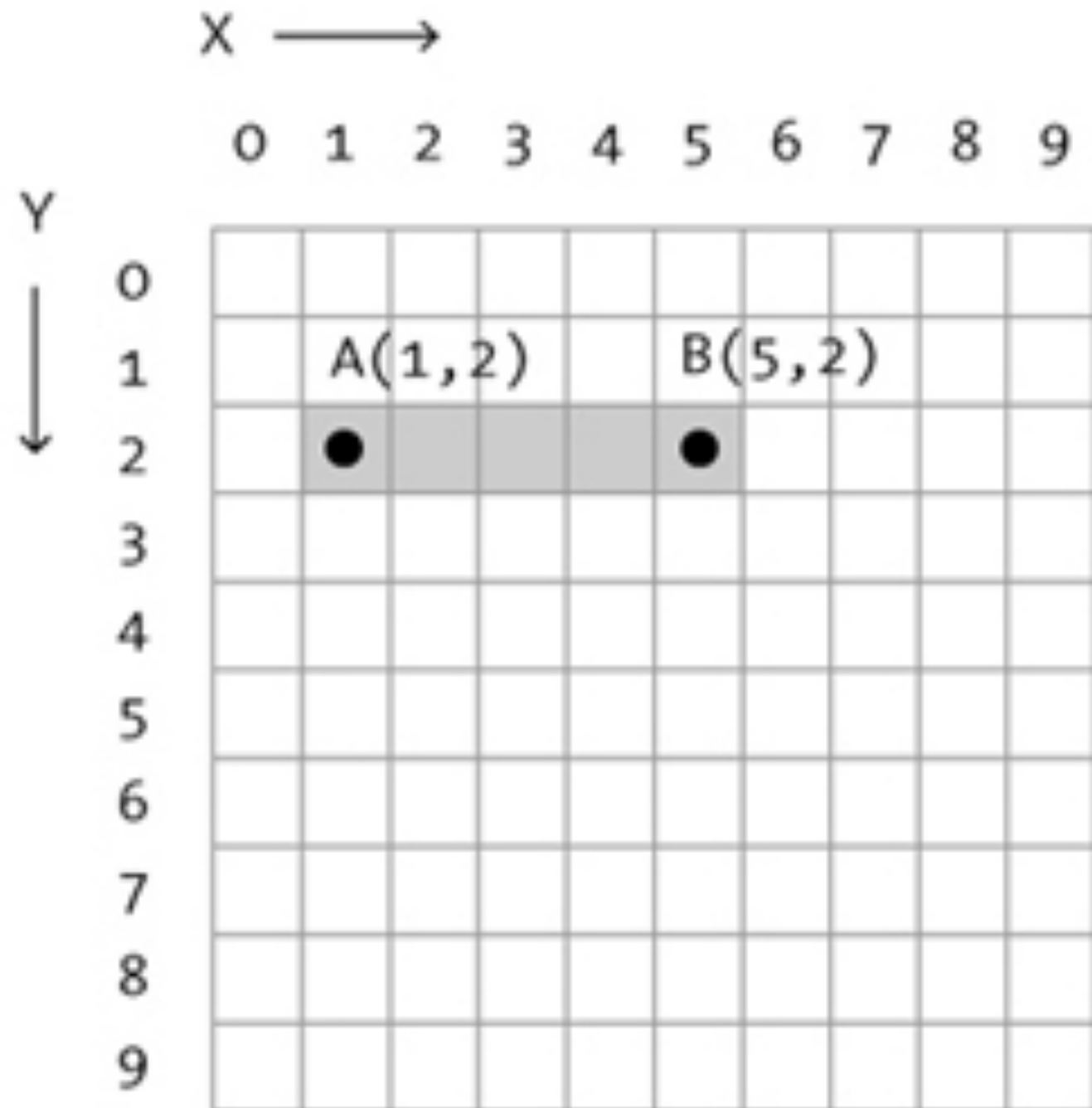
Intro to Processing

- a line
 - line (1,2,5,2)

<http://processing.org/>

Intro to Processing

- a **line**
 - line (1,2,5,2)



<http://processing.org/>

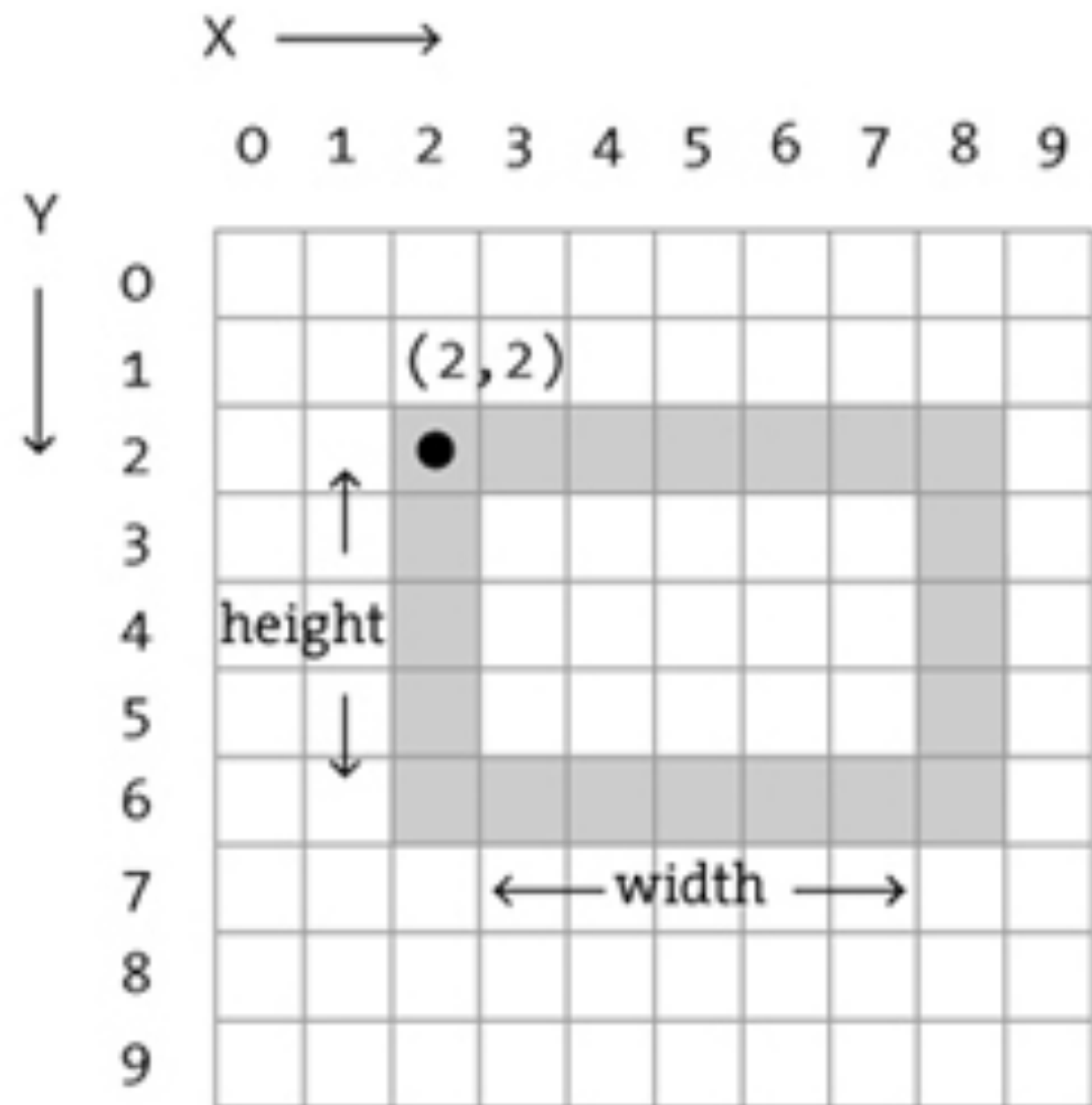
Intro to Processing

- a rectangle
 - `rect (2,2,7,5)`

<http://processing.org/>

Intro to Processing

- a rectangle
 - `rect(2,2,7,5)`

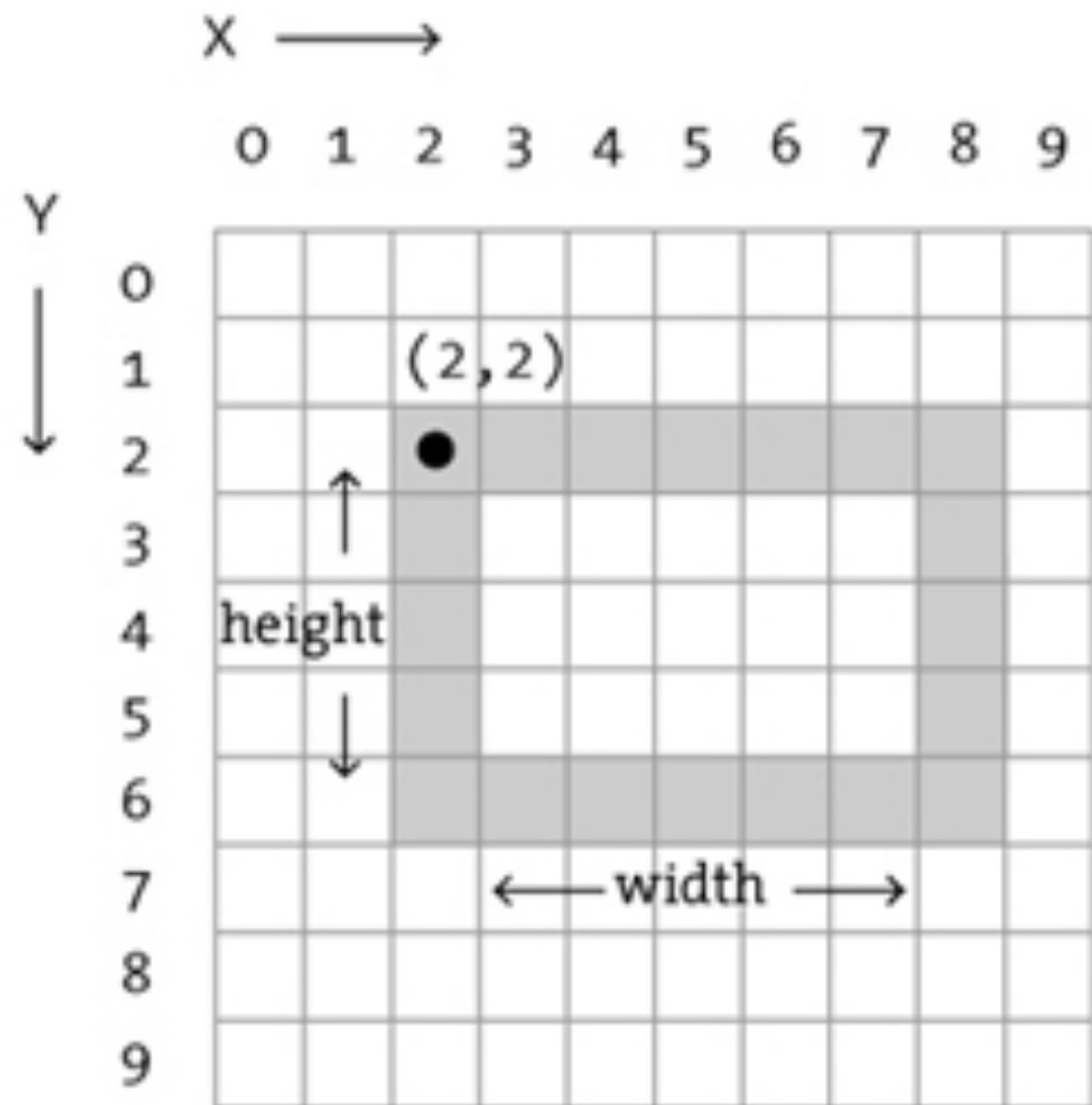


<http://processing.org/>

Intro to Processing

- a rectangle
 - `rect (2,2,7,5)`

Watch Out! 7
is width and 5
is height!



<http://processing.org/>

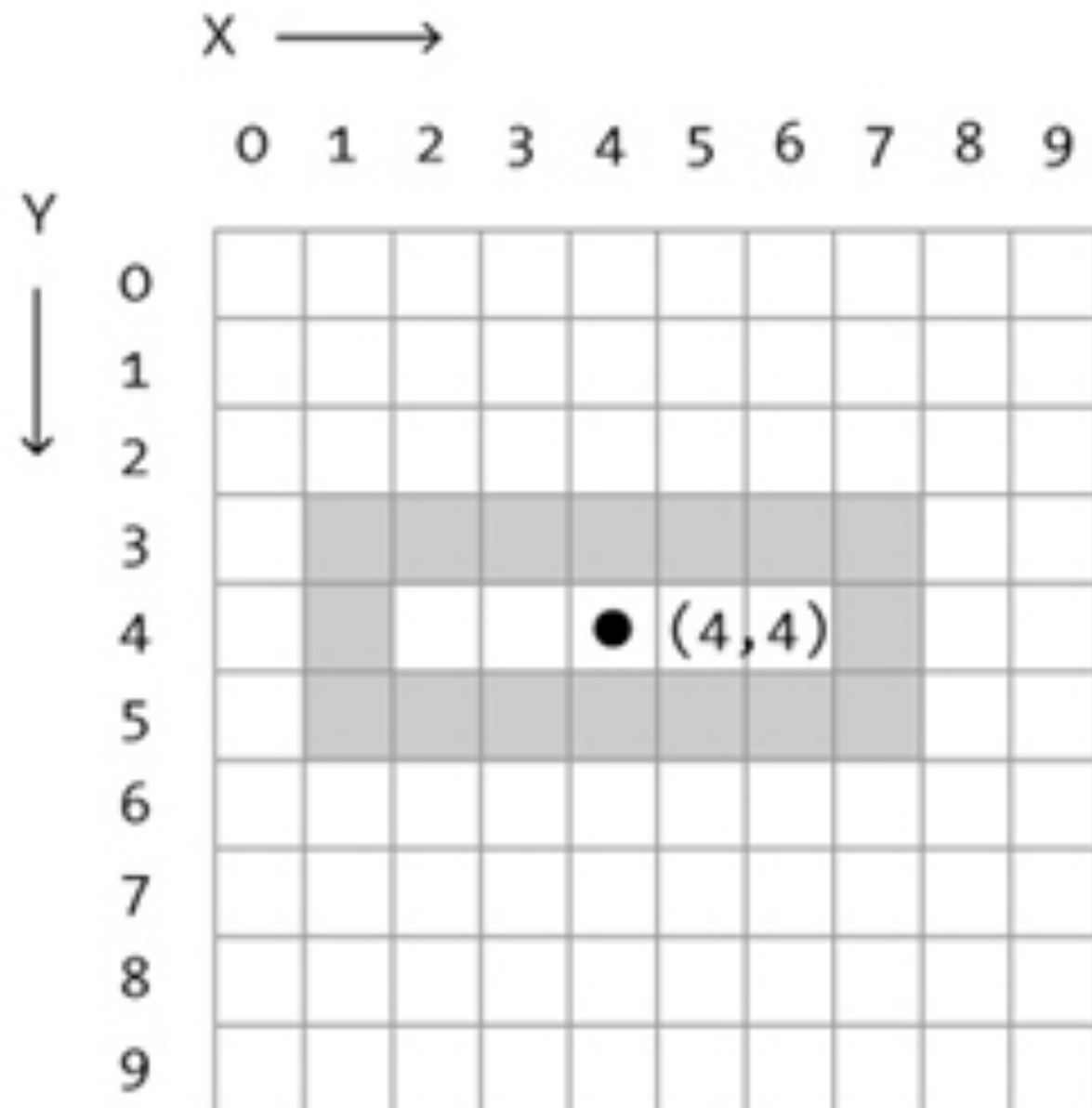
Intro to Processing

- a **rectangle**
 - `rectMode(CENTER)`
 - `rect (4,4,7,3)`

<http://processing.org/>

Intro to Processing

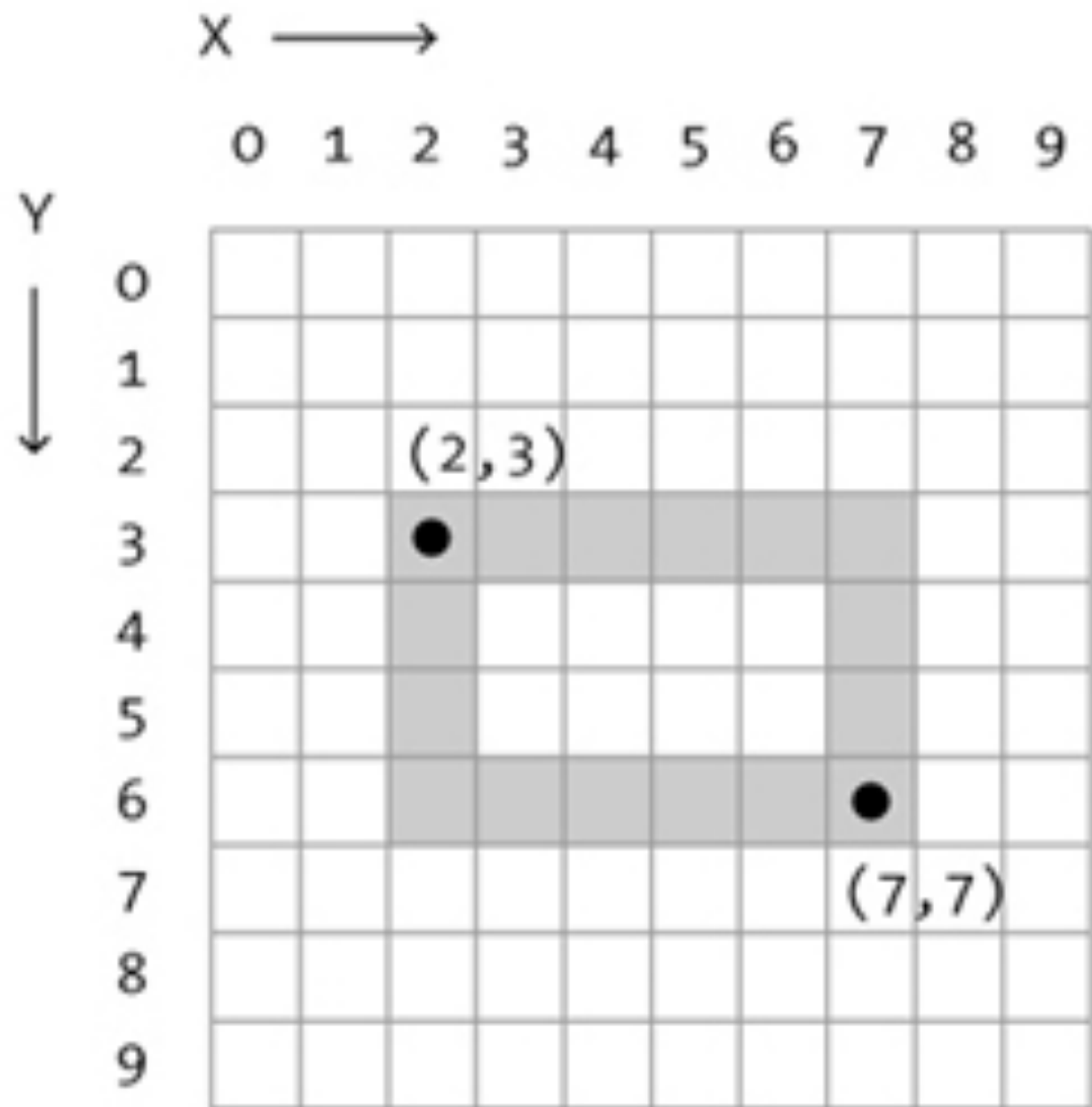
- a **rectangle**
 - `rectMode(CENTER)`
 - `rect(4,4,7,3)`



<http://processing.org/>

Intro to Processing

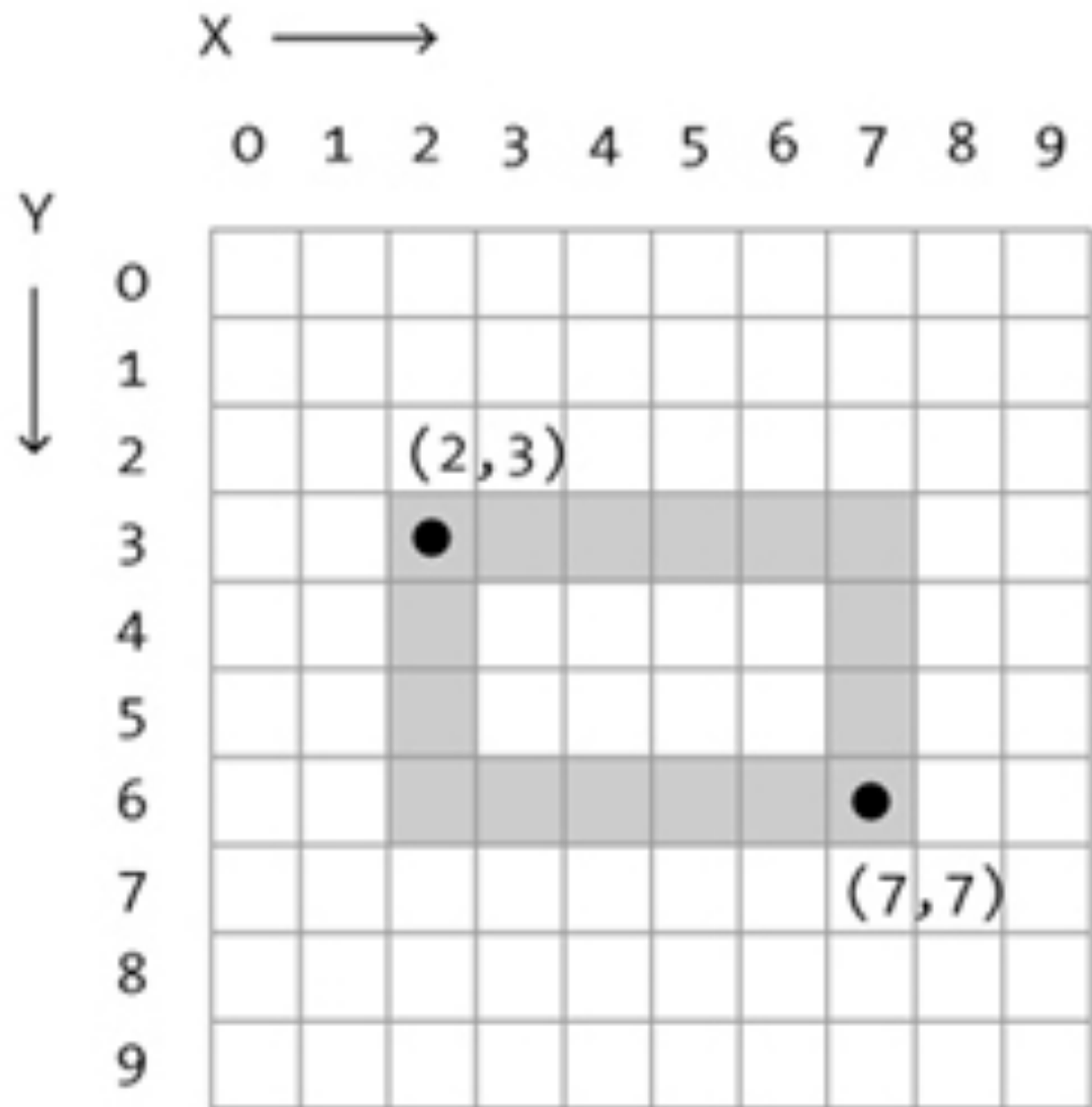
- a **rectangle**
 - rectMode(CORNER)
 - rect (2,3,7,7)



<http://processing.org/>

Intro to Processing

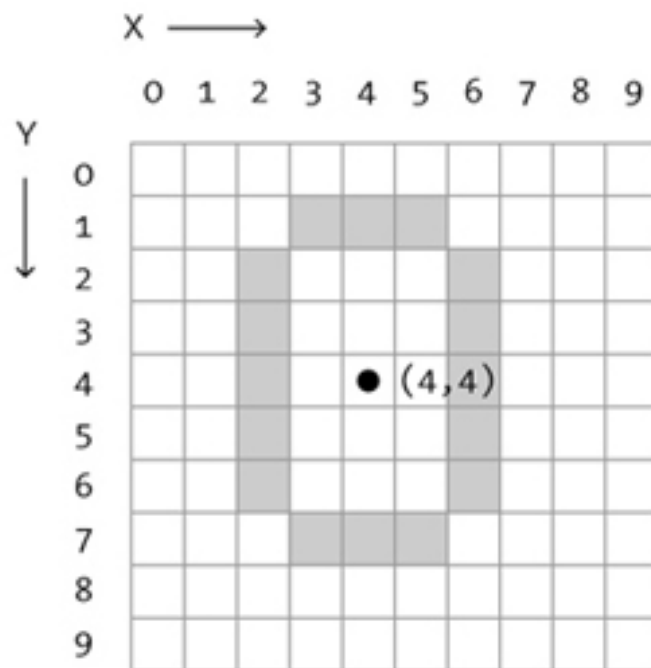
- a **rectangle**
 - rectMode(CORNER)
 - rect (2,3,7,7)



<http://processing.org/>

Intro to Processing

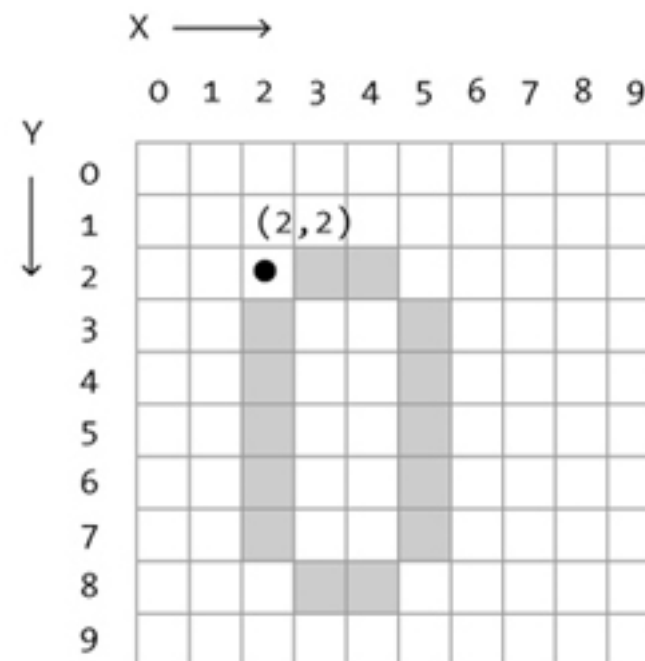
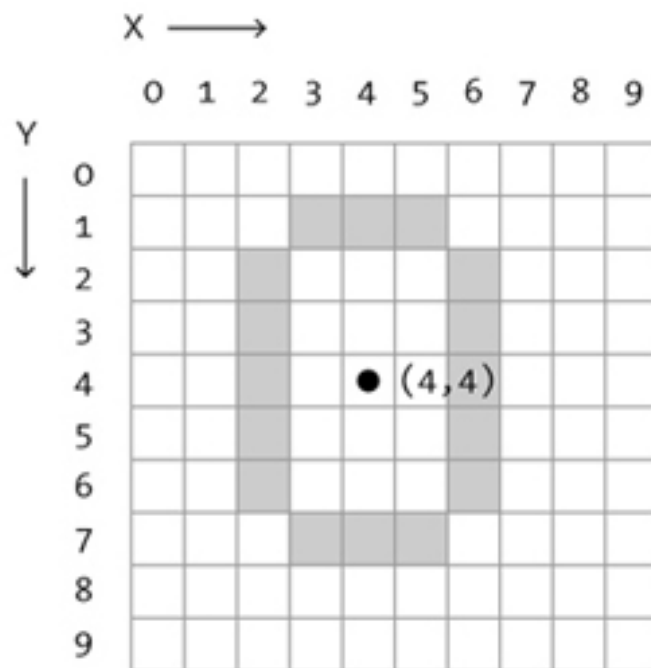
- an **ellipse**
 - `ellipseMode(CENTER)`
 - `ellipse(4,4,5,7)`



<http://processing.org/>

Intro to Processing

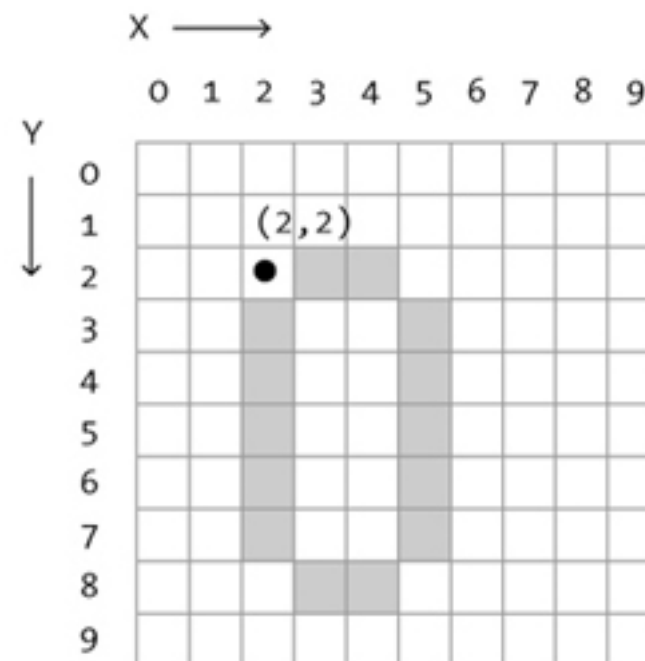
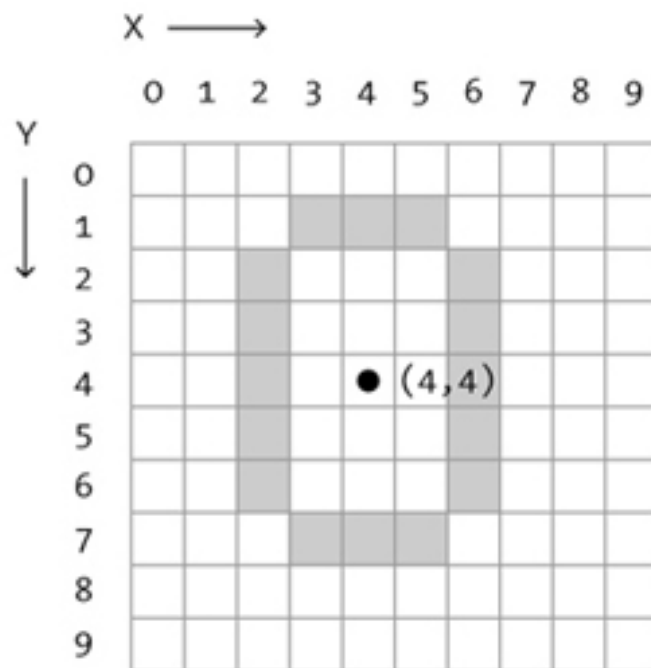
- an ellipse
 - ellipseMode(CENTER)
 - ellipse(4,4,5,7)
- an ellipse
 - ellipseMode(CORNER)
 - ellipse(2,2,4,7)



<http://processing.org/>

Intro to Processing

- an ellipse
 - ellipseMode(CENTER)
 - ellipse(4,4,5,7)
- an ellipse
 - ellipseMode(CORNER)
 - ellipse(2,2,4,7)



Watch Out! 4
is width and 7
is height!

[http](http://www.processing.org)

Intro to Processing

- an **ellipse**

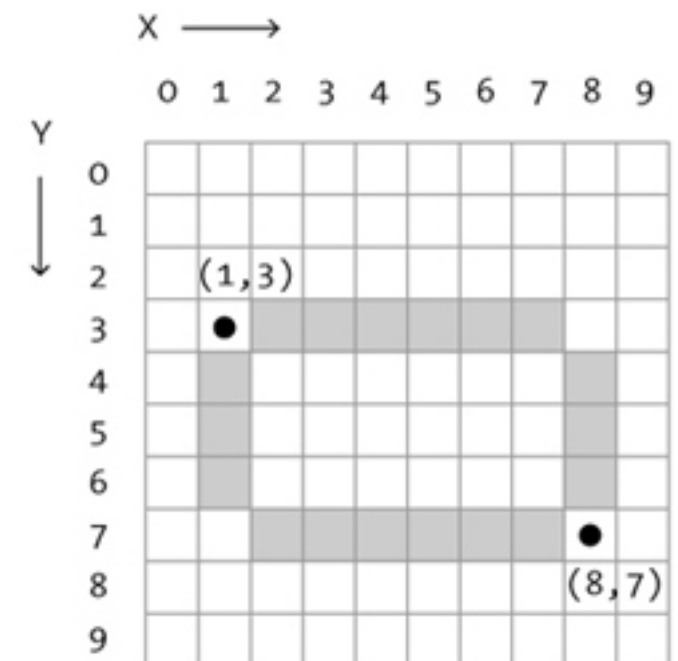
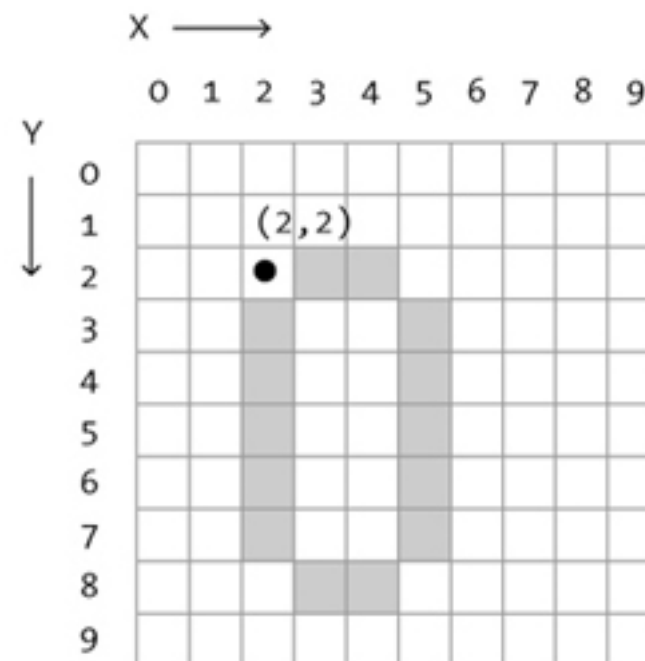
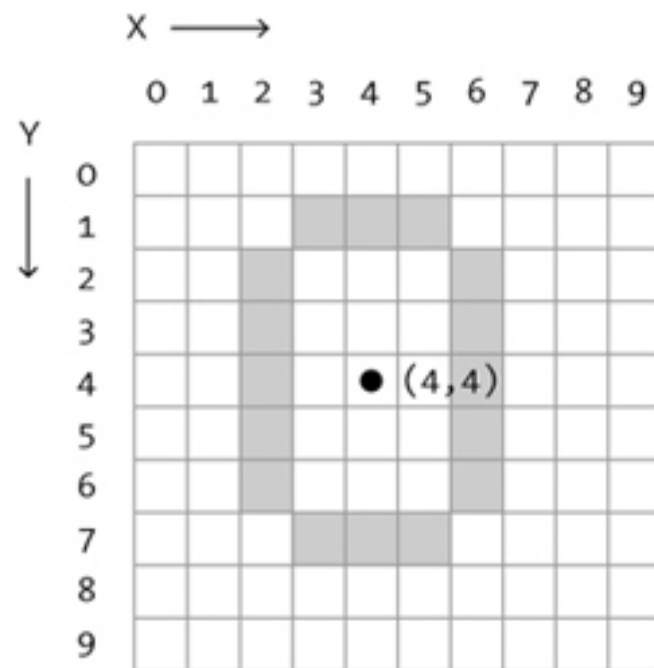
- ellipseMode(CENTER)
- ellipse(4,4,5,7)

- an **ellipse**

- ellipseMode(CORNER)
- ellipse(2,2,4,7)

- an **ellipse**

- ellipseMode(CORNERS)
- ellipse(1,3,8,7)

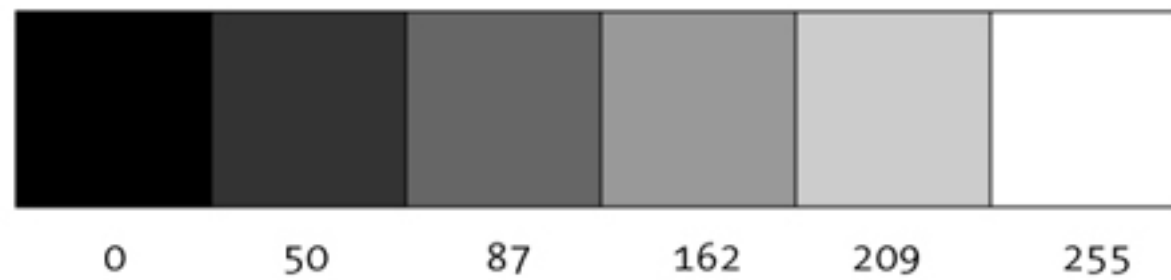


Watch Out! 4
is width and 7
is height!

[http](http://www.processing.org)

Intro to Processing

- when describing color to a computer you must be precise
 - Grayscale color



<http://processing.org/>

Intro to Processing

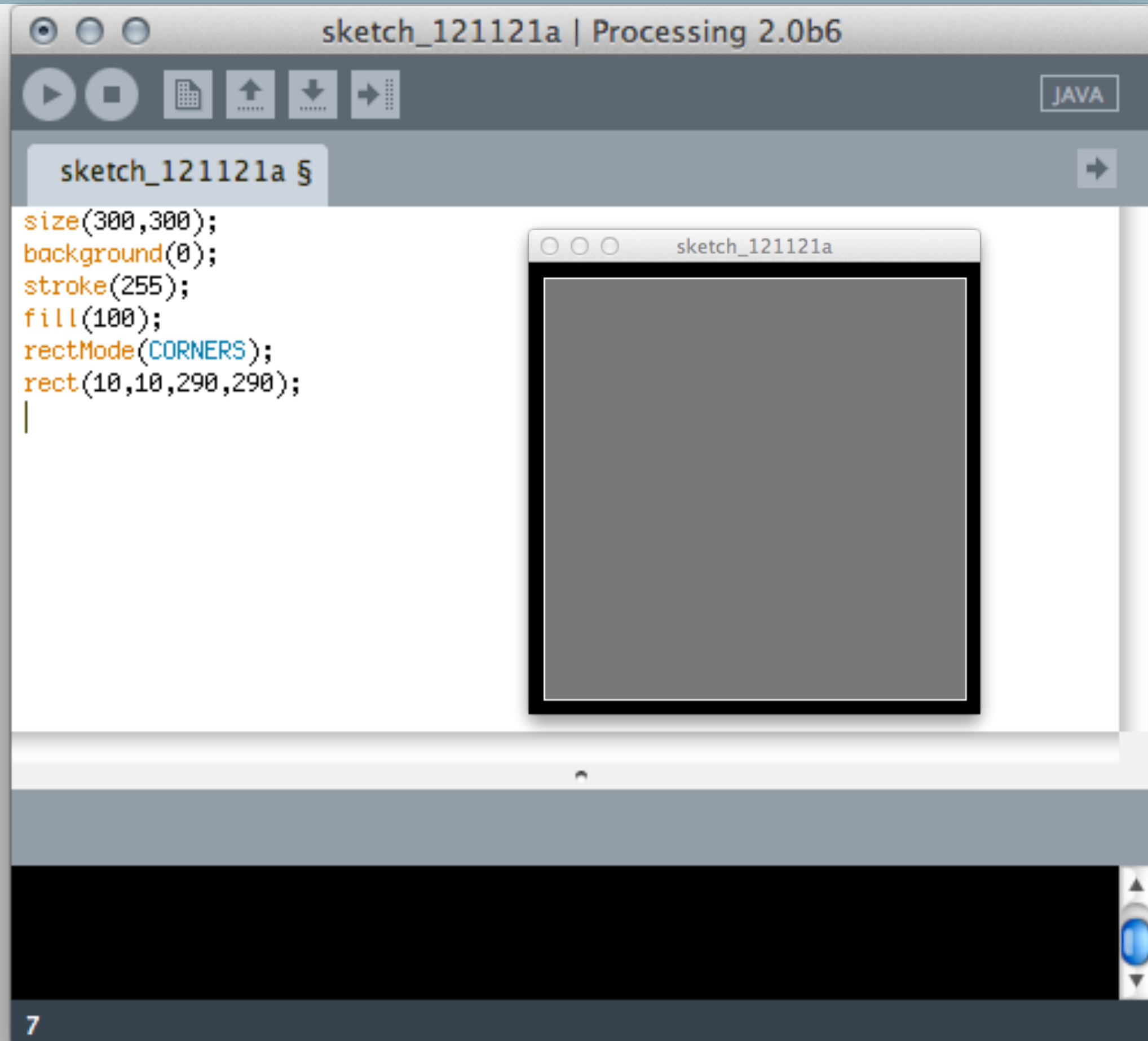
- Three examples of commands that use color
 - Set the background color
 - `background(<color>);`
 - Pick the pen color that you are going to draw with
 - `stroke(<color>);`
 - Pick the fill color that you are going to draw with
 - `fill(<color>);`

<http://processing.org/>

Intro to Processing



Intro to Processing

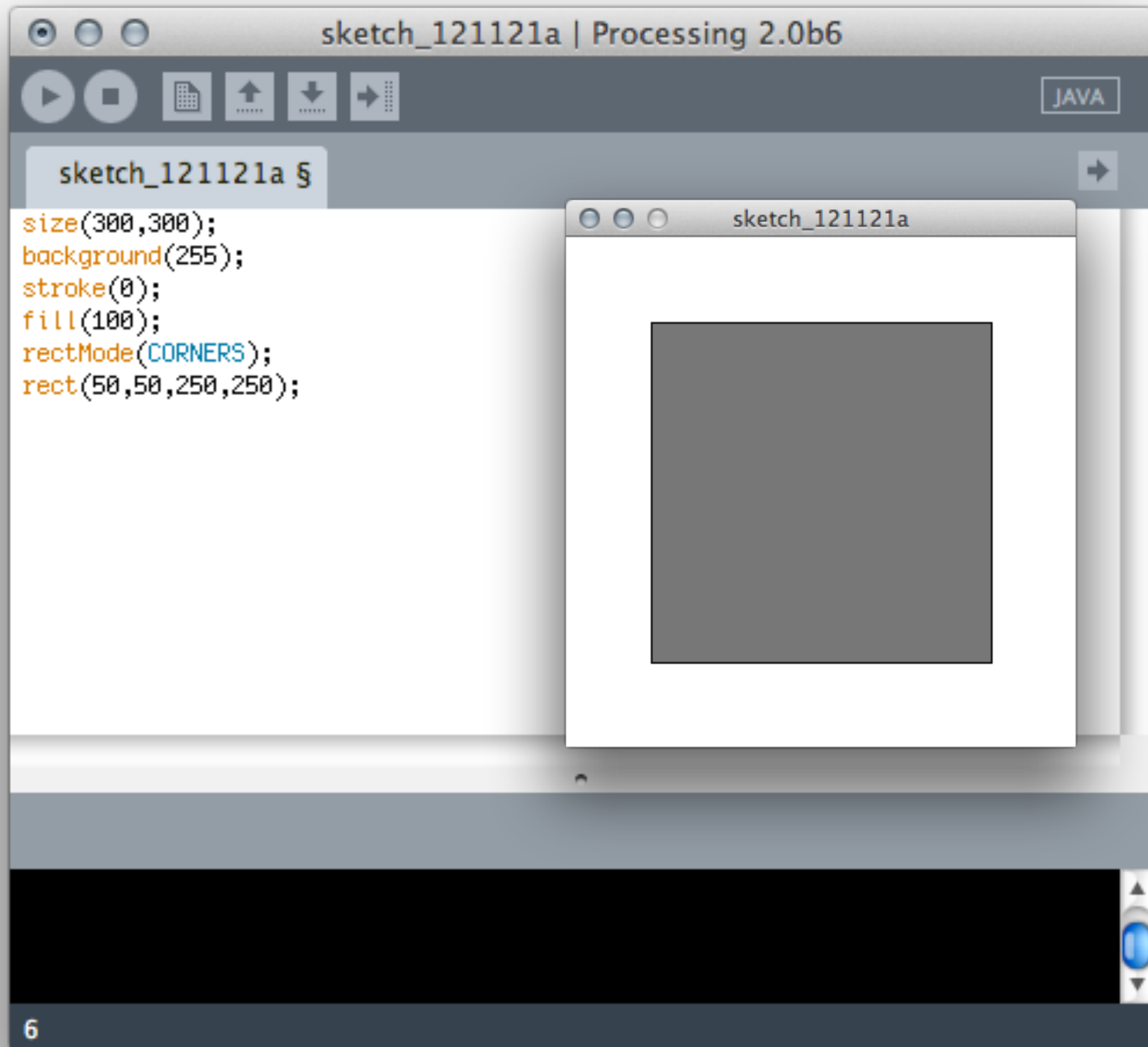


Intro to Processing



<http://processing.org/>

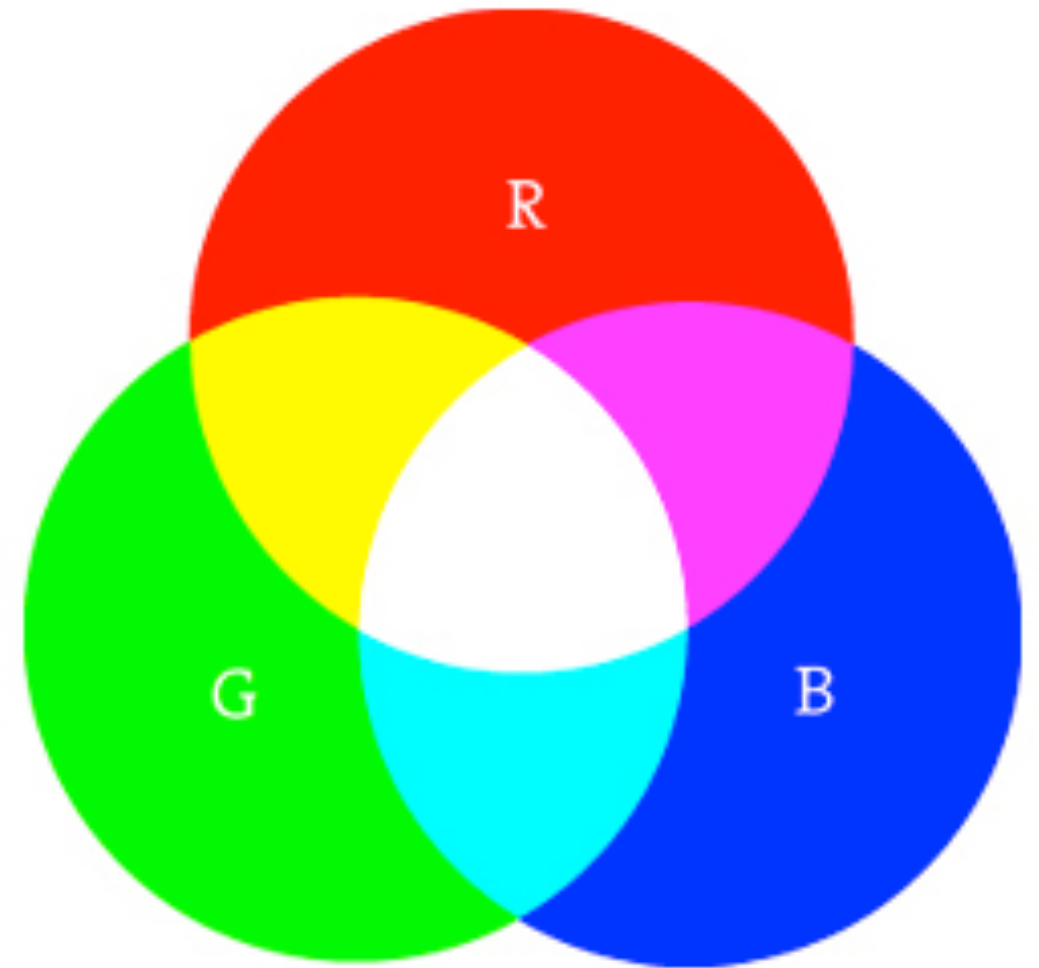
Intro to Processing



<http://processing.org/>

Intro to Processing

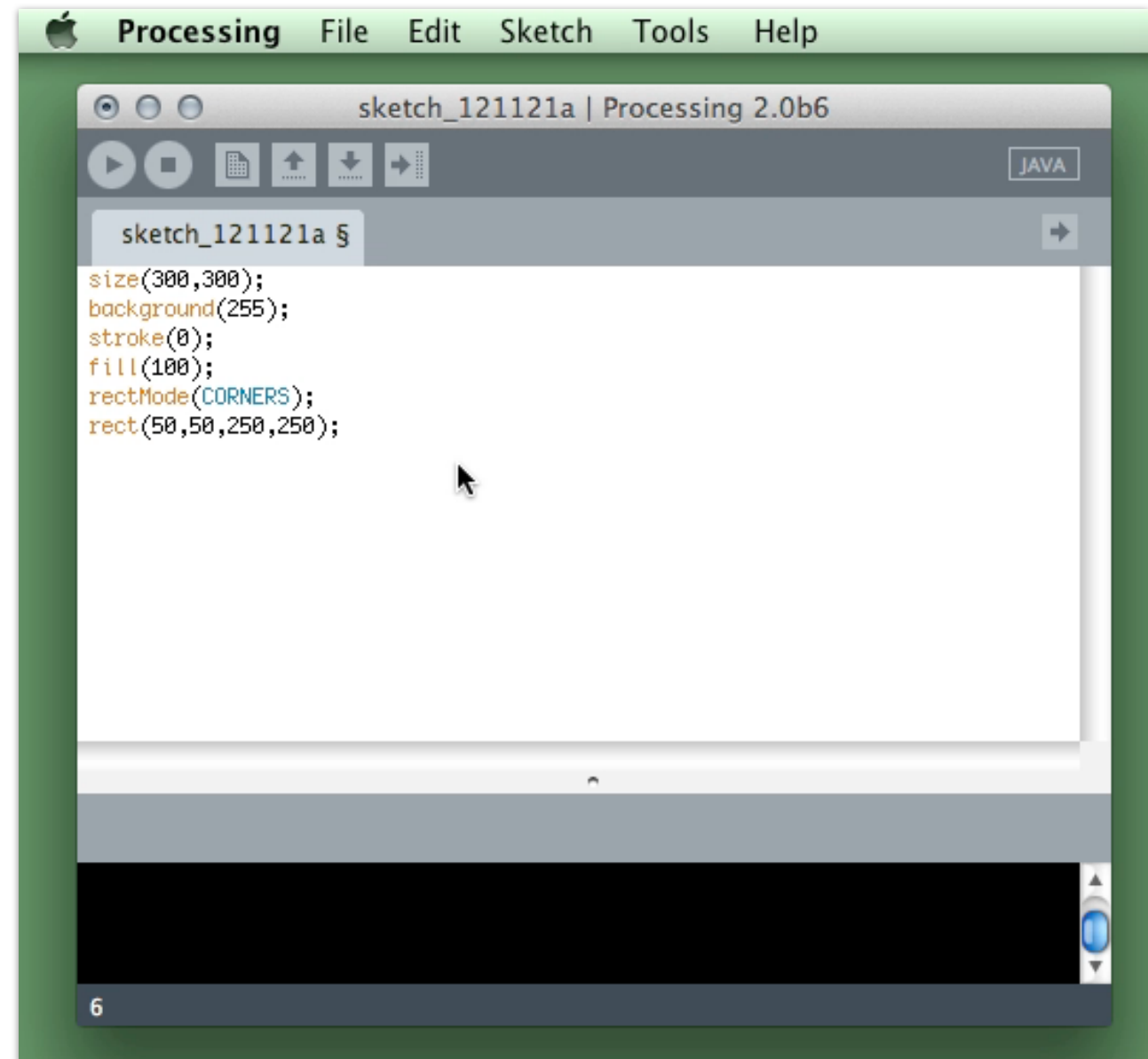
- when describing color to a computer you must be precise
 - RGB color
 - Red
 - Green
 - Blue
 - Red + Green = Yellow
 - Green + Blue = Cyan
 - Red + Blue = Magenta
 - Red + Green + Blue = White
 - no color = Black



<http://processing.org/>

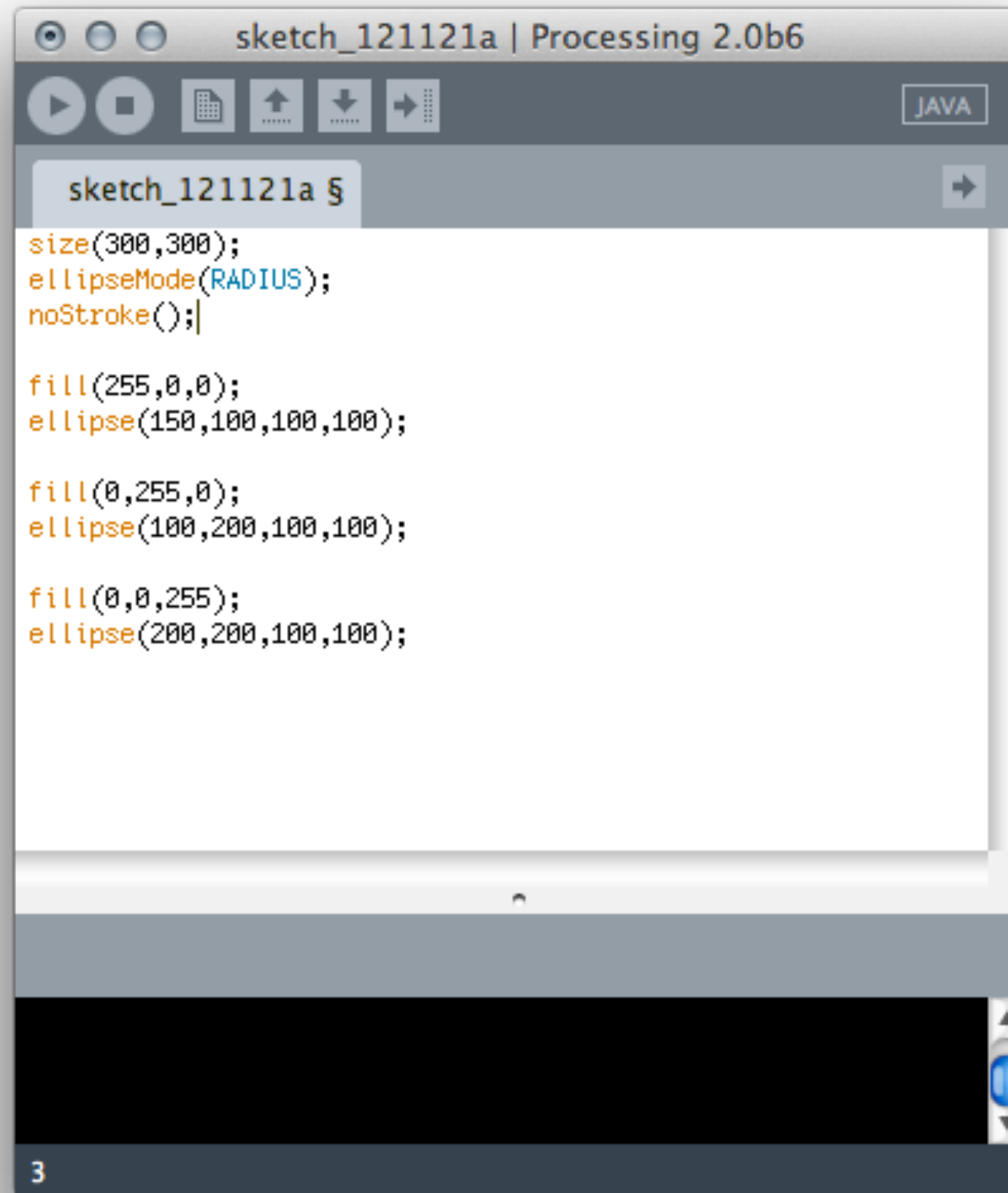
Intro to Processing

- when describing color to a computer you must be precise
 - RGB color
 - Red
 - Green
 - Blue
 - Red + Green = Yellow
 - Green + Blue = Cyan
 - Red + Blue = Magenta
 - Red + Green + Blue = White
 - no color = Black



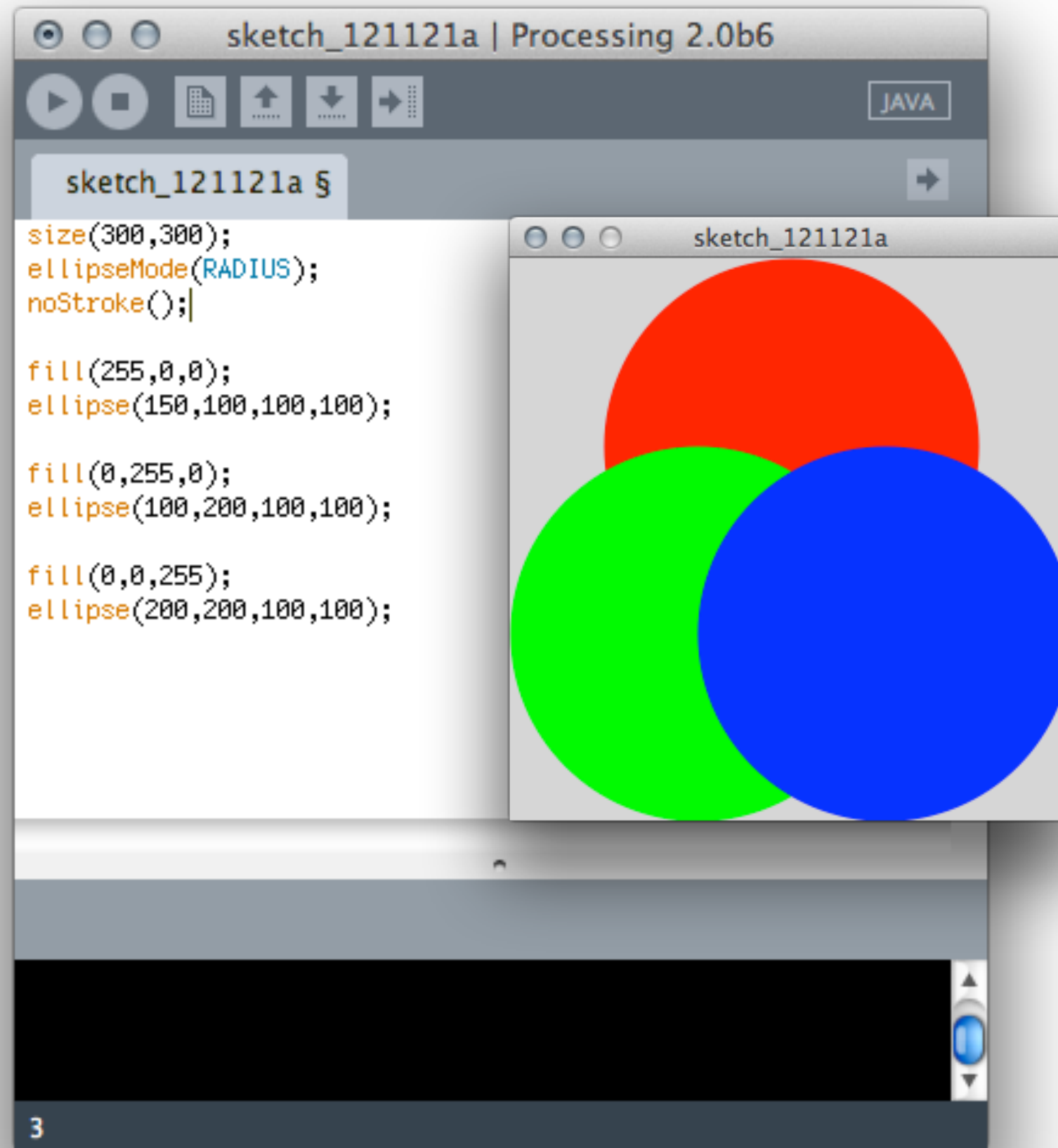
<http://processing.org/>

Intro to Processing



<http://processing.org/>

Intro to Processing



<http://processing.org/>

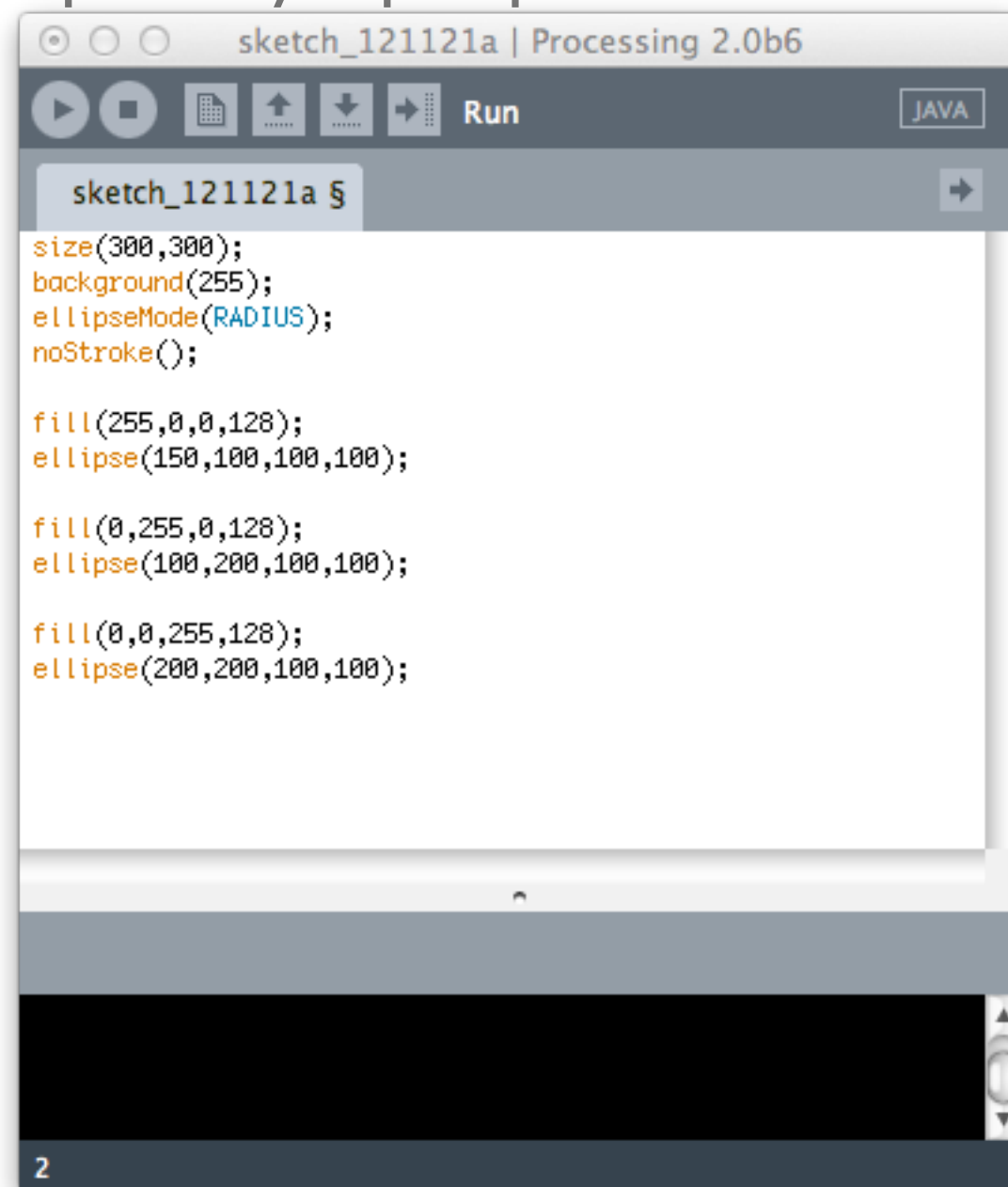
Intro to Processing

- when describing color to a computer you must be precise
 - alpha transparency
 - allows for colors to blend when on top of each other
 - 0 is completely transparent
 - 255 is completely opaque

<http://processing.org/>

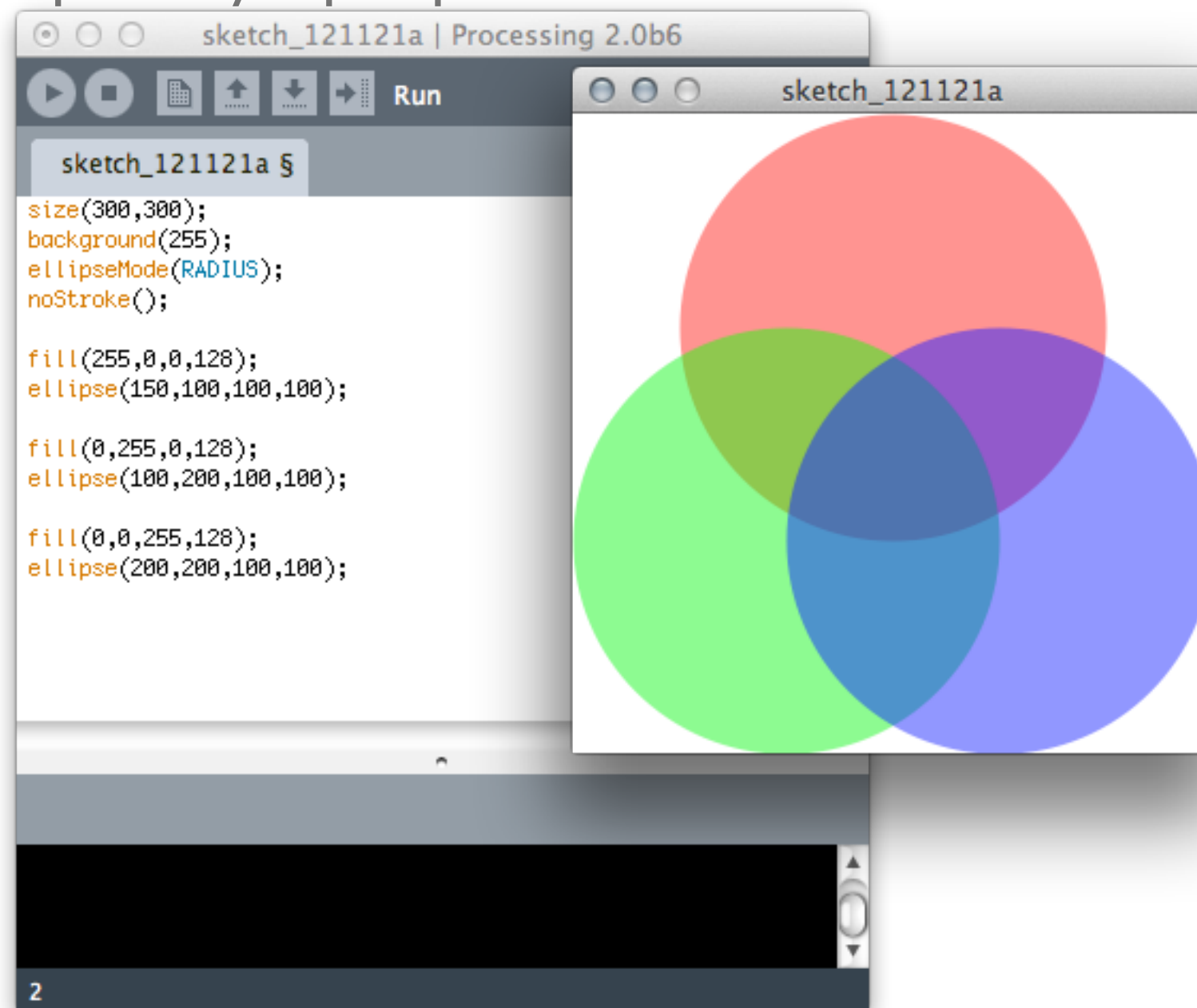
Intro to Processing

- when describing color to a computer you must be precise
 - **alpha transparency**
 - allows for colors to blend when on top of each other
 - **0** is completely transparent
 - **255** is completely opaque



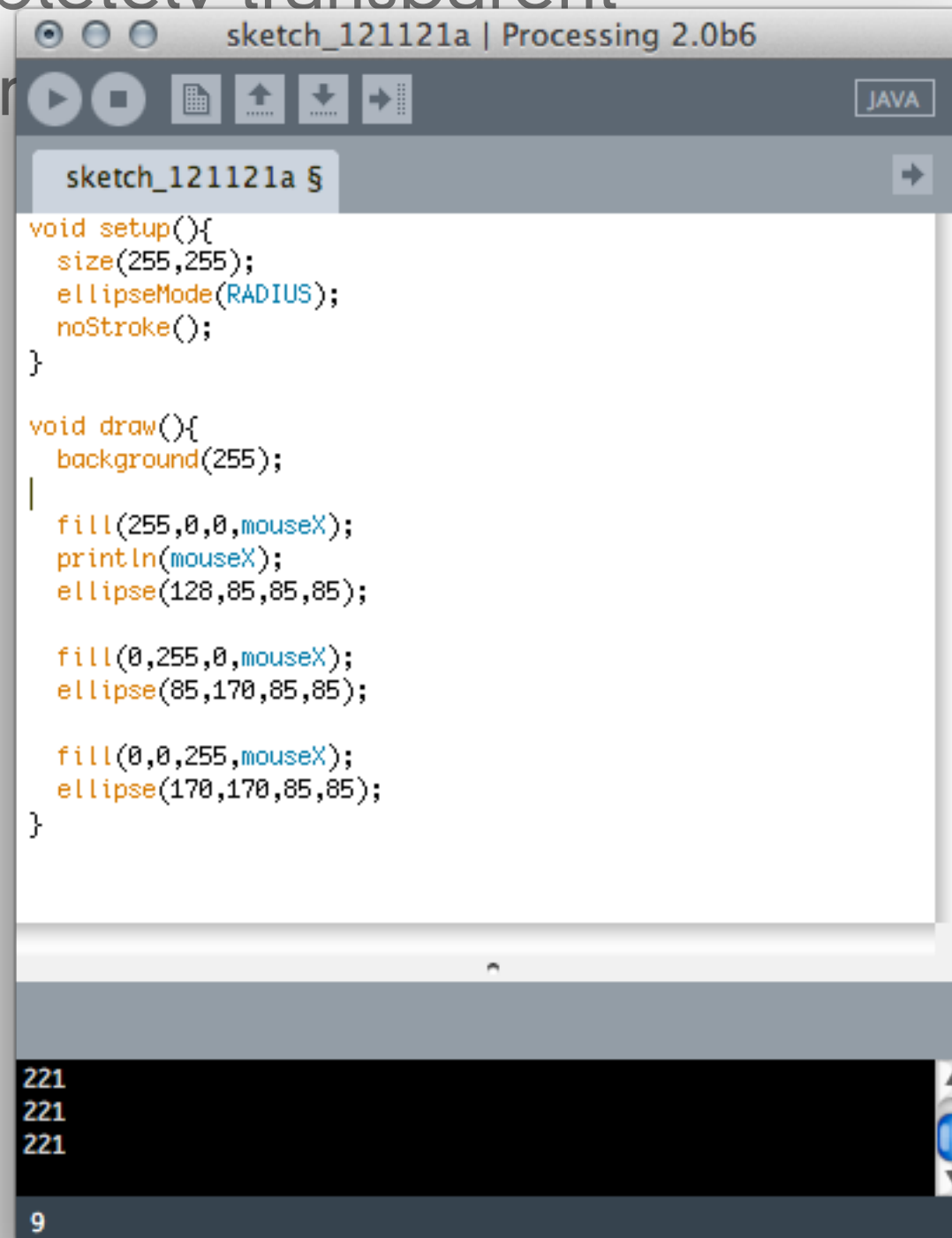
Intro to Processing

- when describing color to a computer you must be precise
 - **alpha transparency**
 - allows for colors to blend when on top of each other
 - **0** is completely transparent
 - **255** is completely opaque



Intro to Processing

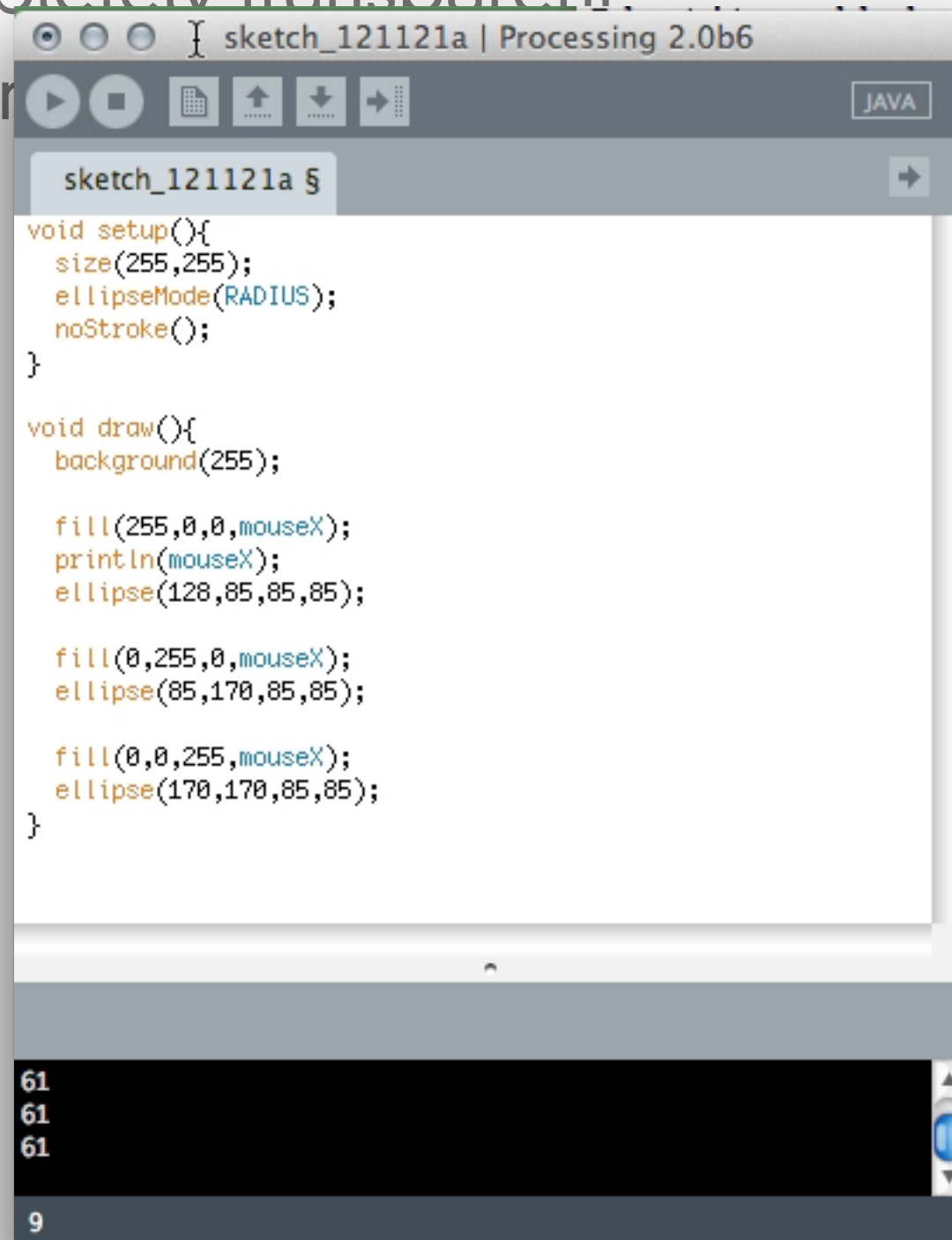
- when describing color to a computer you must be precise
 - **alpha transparency**
 - allows for colors to blend when on top of each other
 - **0** is completely transparent
 - **255** is completely opaque



```
sketch_121121a | Processing 2.0b6  
sketch_121121a $  
void setup(){  
  size(255,255);  
  ellipseMode(RADIUS);  
  noStroke();  
}  
  
void draw(){  
  background(255);  
  |  
  fill(255,0,0,mouseX);  
  println(mouseX);  
  ellipse(128,85,85,85);  
  
  fill(0,255,0,mouseX);  
  ellipse(85,170,85,85);  
  
  fill(0,0,255,mouseX);  
  ellipse(170,170,85,85);  
}  
  
221  
221  
221  
9
```

Intro to Processing

- when describing color to a computer you must be precise
 - **alpha transparency**
 - allows for colors to blend when on top of each other
 - **0** is completely transparent
 - **255** is completely opaque



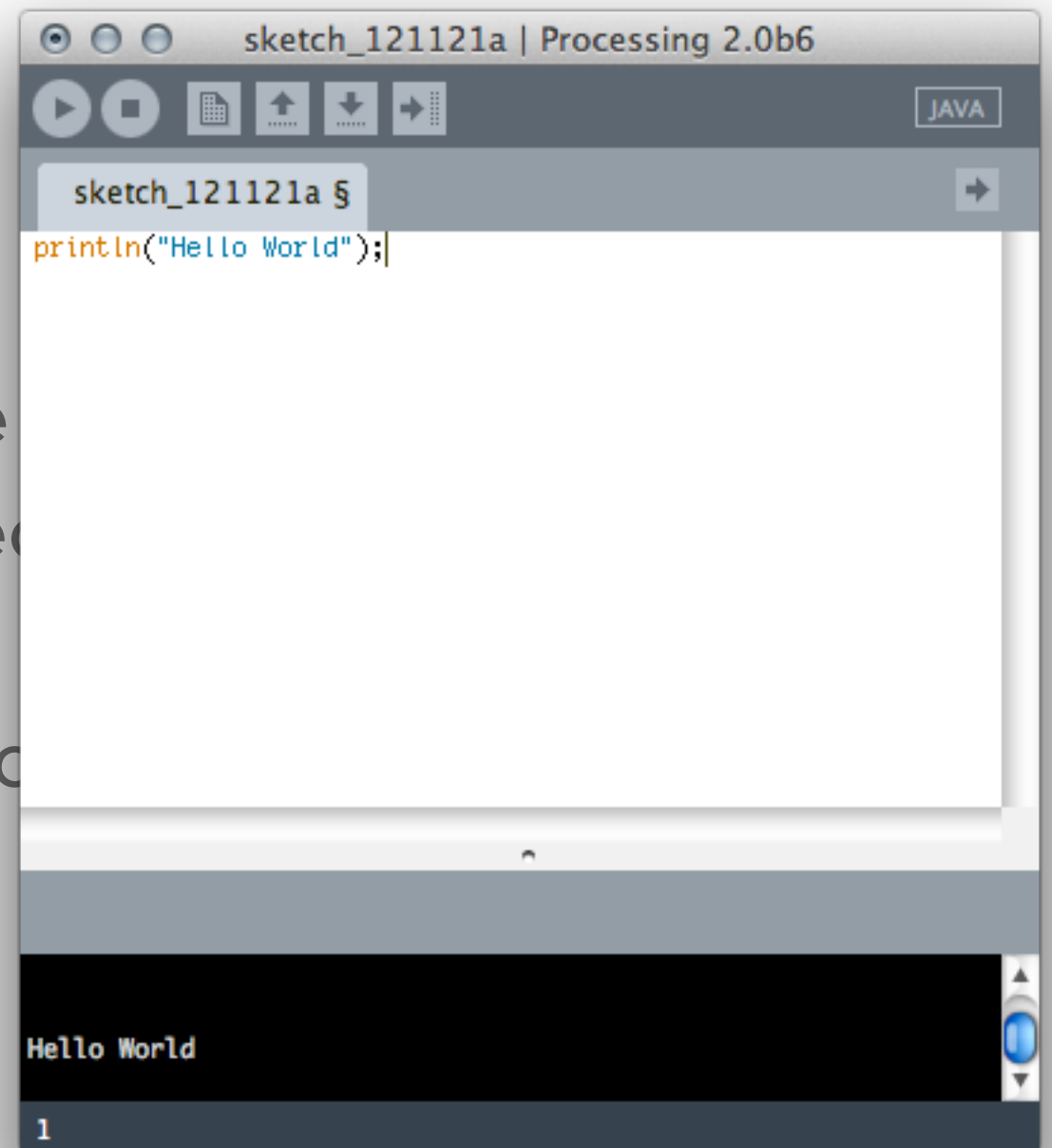
```
sketch_121121a | Processing 2.0b6  
sketch_121121a §  
void setup(){  
  size(255,255);  
  ellipseMode(RADIUS);  
  noStroke();  
}  
  
void draw(){  
  background(255);  
  
  fill(255,0,0,mouseX);  
  println(mouseX);  
  ellipse(128,85,85,85);  
  
  fill(0,255,0,mouseX);  
  ellipse(85,170,85,85);  
  
  fill(0,0,255,mouseX);  
  ellipse(170,170,85,85);  
}  
  
61  
61  
61  
9
```

- Text
 - String
 - A collection of letters that are put between quotes
 - They get treated as one object
 - "Hello World" is a String
 - Displaying text in the message console
 - `println("Hello World")`

<http://processing.org/>

Intro to Processing

- Text
 - **String**
 - A collection of letters that are
 - They get treated as one object
 - “Hello World” is a String
 - Displaying text in the message window
 - `println(“Hello World”)`



<http://processing.org/>

Intro to Processing

- Displaying text in a sketch
 - Find out what fonts are available to you
 - `PFont.list()`

<http://processing.org/>

Intro to Processing

- Displaying text in a sketch
 - Find out what fonts are available to you
 - `PFont.list()`

A screenshot of the Processing IDE window titled "sketch_121121a | Processing 2.0b6". The code editor shows the command `println(PFont.list());`. The output window displays a list of 14 fonts as an array: `[0] "Serif"`, `[1] "SansSerif"`, `[2] "Monospaced"`, `[3] "Dialog"`, `[4] "DialogInput"`, `[5] "ACaslonPro-Bold"`, `[6] "ACaslonPro-BoldItalic"`, `[7] "ACaslonPro-Italic"`, `[8] "ACaslonPro-Regular"`, `[9] "ACaslonPro-Semibold"`, `[10] "ACaslonPro-SemiboldItalic"`, `[11] "AGaramondPro-Bold"`, `[12] "AGaramondPro-BoldItalic"`, and `[13] "AGaramondPro-Italic"`. The line number 1 is visible at the bottom of the output window.

```
sketch_121121a §  
println(PFont.list());  
  
[0] "Serif"  
[1] "SansSerif"  
[2] "Monospaced"  
[3] "Dialog"  
[4] "DialogInput"  
[5] "ACaslonPro-Bold"  
[6] "ACaslonPro-BoldItalic"  
[7] "ACaslonPro-Italic"  
[8] "ACaslonPro-Regular"  
[9] "ACaslonPro-Semibold"  
[10] "ACaslonPro-SemiboldItalic"  
[11] "AGaramondPro-Bold"  
[12] "AGaramondPro-BoldItalic"  
[13] "AGaramondPro-Italic"  
1
```

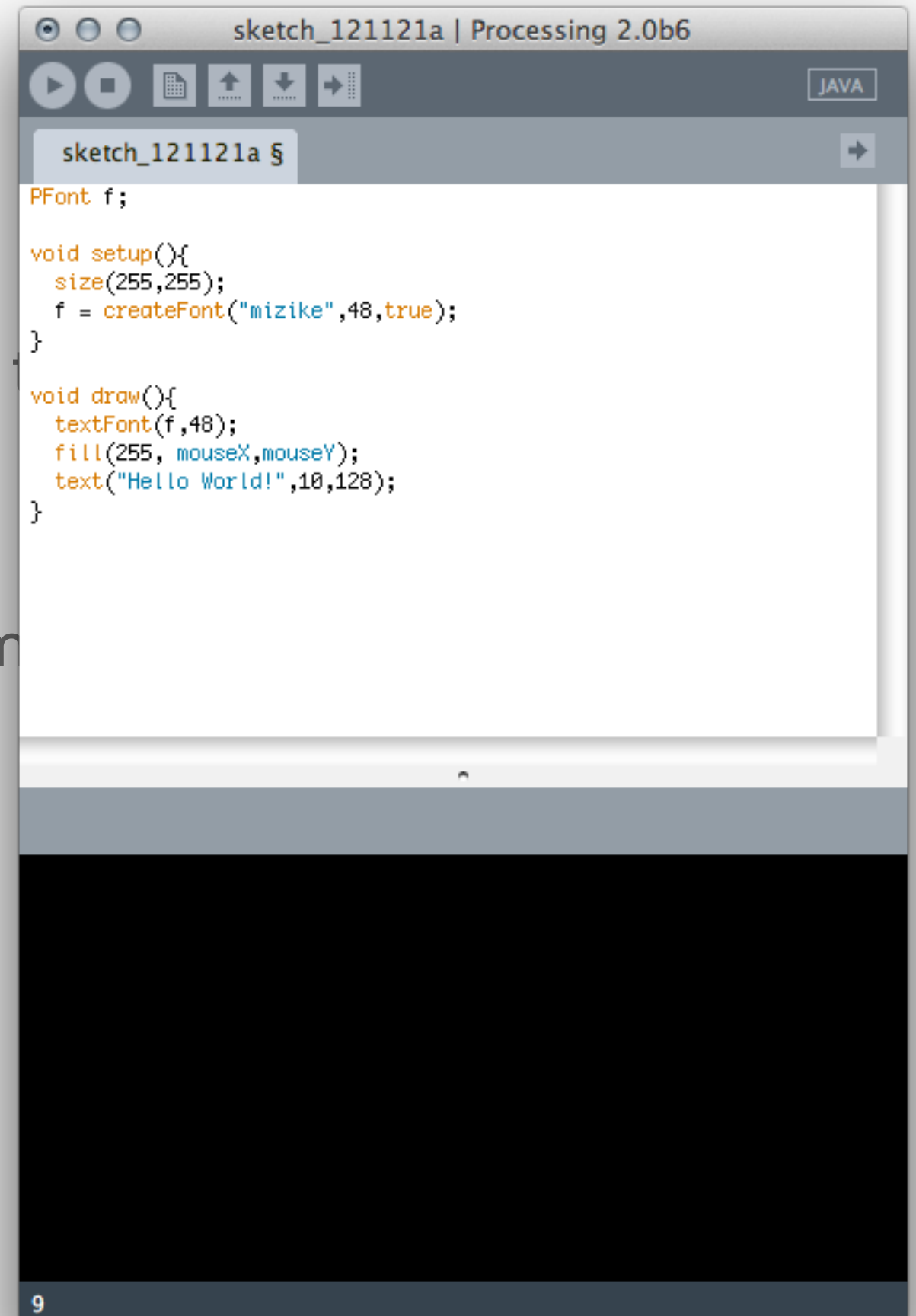
<http://processing.org/>

- Create a font to use in `setup()`
 - `createFont()`
- Specify the way to write the font to in `draw()`
 - `textFont()`
 - `fill()`
- Specify the string and the placement in `draw()`
 - `text("Hello World!", 50, 50)`

<http://processing.org/>

Intro to Processing

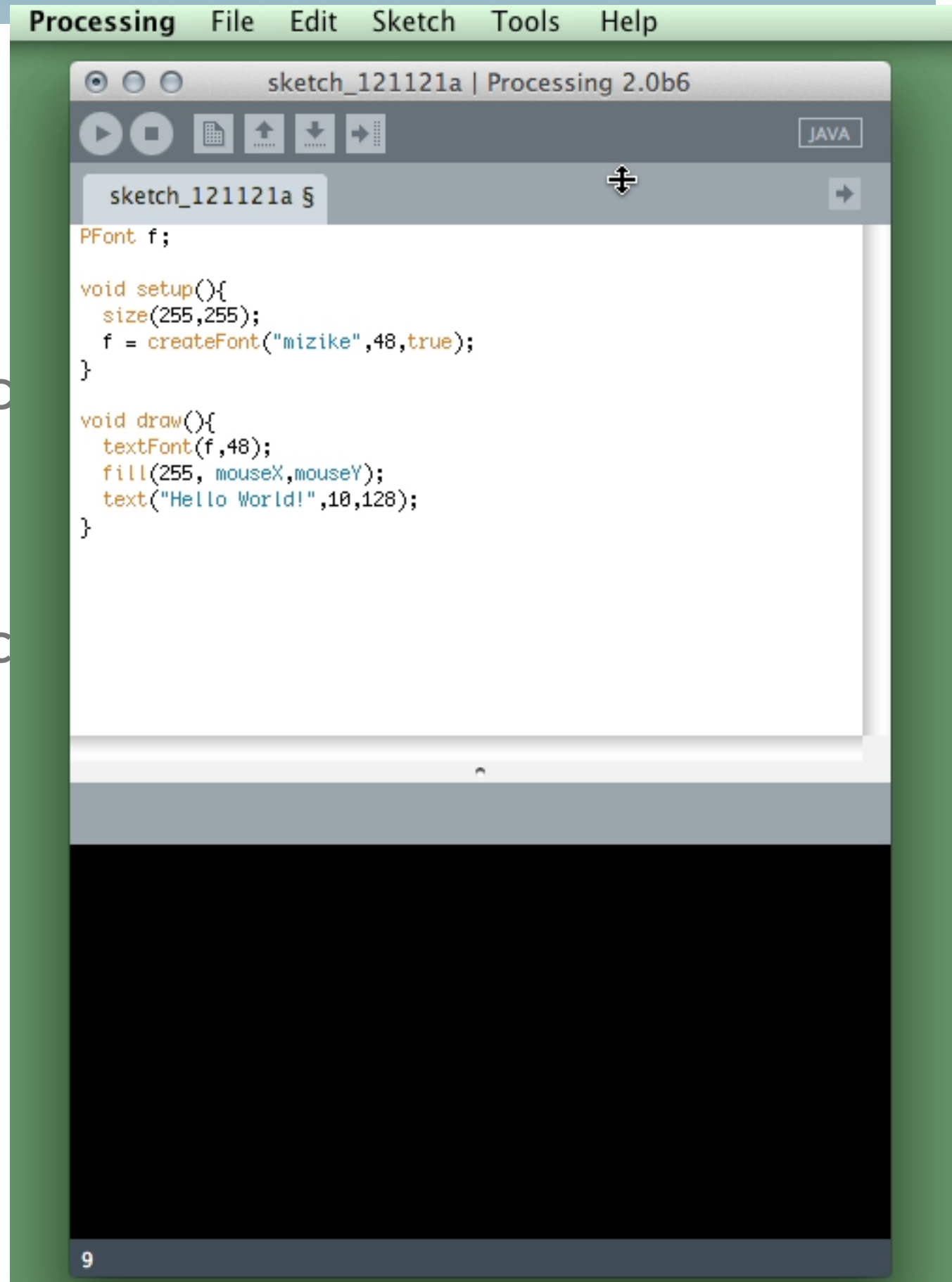
- Create a font to use in `setup()`
 - `createFont()`
- Specify the way to write the font
 - `textFont()`
 - `fill()`
- Specify the string and the placement
 - `text("Hello World!", 50, 50)`



<http://processing.org/>

Intro to Processing

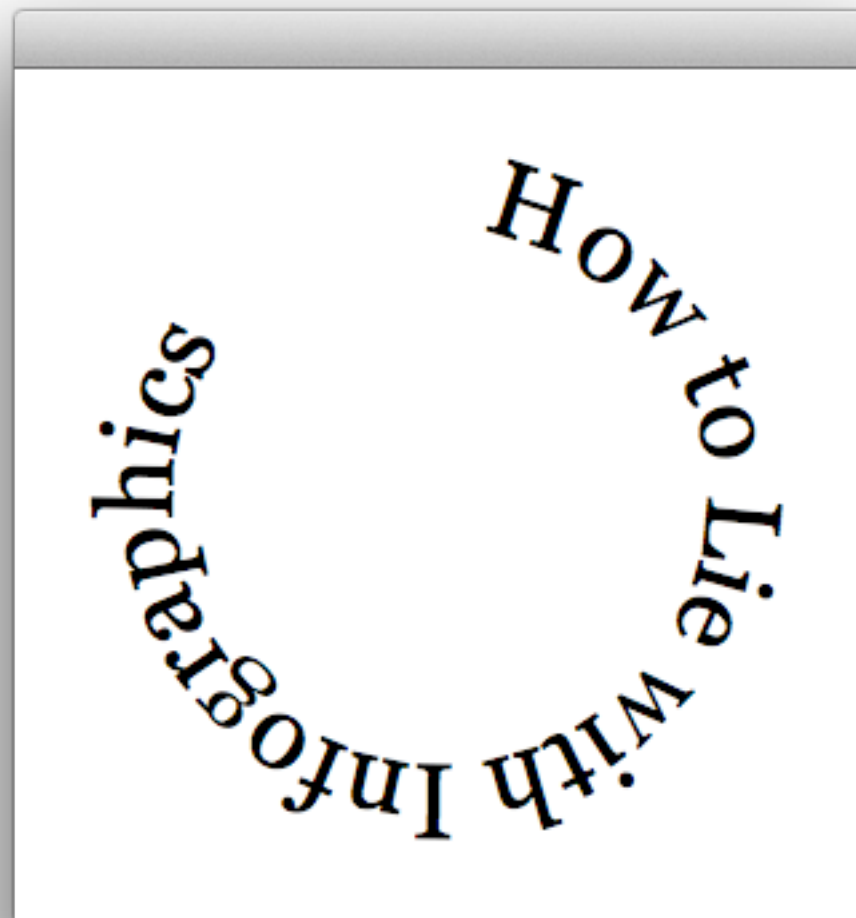
- Create a font to use in `setup()`
 - `createFont()`
- Specify the way to write the font
 - `textFont()`
 - `fill()`
- Specify the string and the place to write the text
 - `text("Hello World!", 50, 50)`



<http://processing.org/>

Intro to Processing

- Going nuts!



```
void setup() {
  size(320, 320);
  f = createFont("Georgia", 40, true);
  textFont(f);
  // The text must be centered!
  textAlign(CENTER);
  smooth();
}

void draw() {
  background(255);

  // Start in the center and draw the circle
  translate(width / 2, height / 2);
  noFill();
  stroke(0);

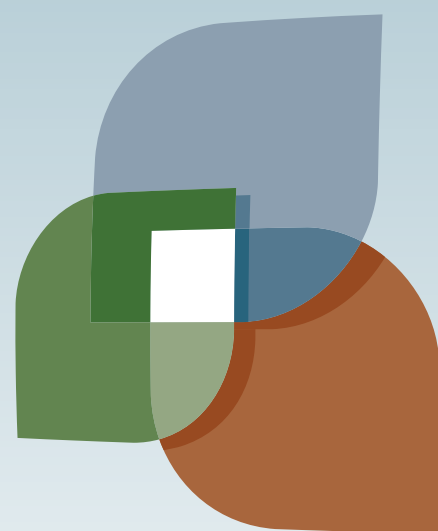
  // We must keep track of our position along the curve
  float arclength = 2*mouseX;

  // For every box
  for (int i = 0; i < message.length(); i++)
  {
    // Instead of a constant width, we check the width of each character.
    char currentChar = message.charAt(i);
    float w = textWidth(currentChar);

    // Each box is centered so we move half the width
    arclength += w/2;
    // Angle in radians is the arclength divided by the radius
    // Starting on the left side of the circle by adding PI
    float theta = PI + arclength / r;

    pushMatrix();
    // Polar to cartesian coordinate conversion
    translate(r*cos(theta), r*sin(theta));
    // Rotate the box
    rotate(theta+PI/2); // rotation is offset by 90 degrees
    // Display the character
    fill(0);
    text(currentChar, 0, 0);
    popMatrix();
    // Move halfway again
    arclength += w/2;
  }
}
```

<http://processing.org/>



L U C I

