GRAY AREA
FOUNDATION
FOR THE ARTS

Drawing with Code
Day 1

https://processing.org/download/

WTF is Processing?

- Processing is a programming language created by Casey Reas and Ben Fry
- The language is designed to maximize creativity and minimize coding effort
- Designed for artists, made by computer nerds

What can Processing Do?

Substrate - Jared Tarbell

Phantogram - Fall in Love

How does it work?

- Processing is a programming language
- Processing based on the Java Programming Language
- Processing code can be complied using the Processing IDE

Color By Numbers

Processing is one big robotic Etch-A-Sketch

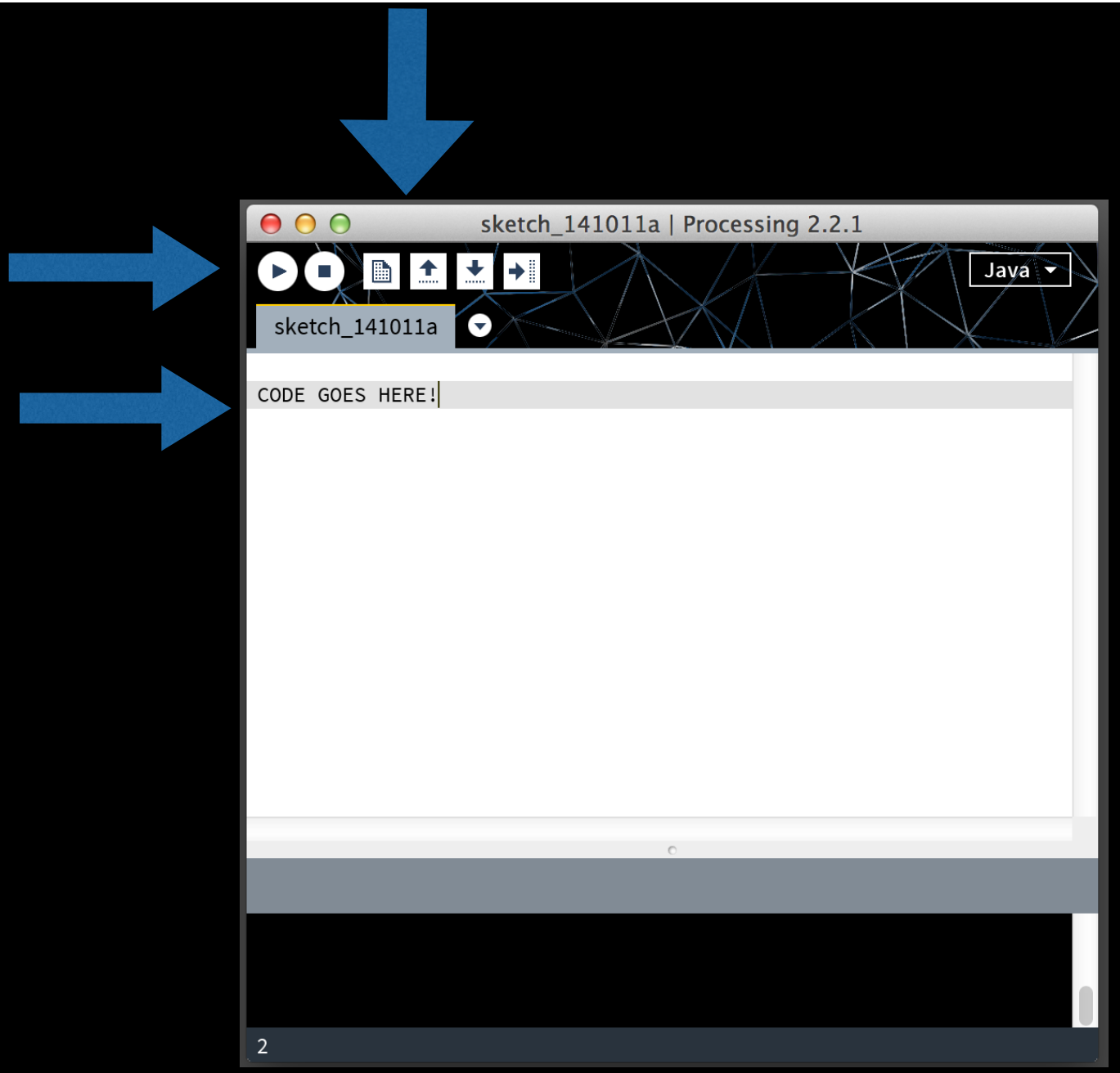
Congrats, You Are All Programmers

GRAY AREA
FOUNDATION
FOR THE ARTS

THE TWO STATES OF EVERY PROGRAMMER

I AM A GOD.

I HAVE NO IDEA WHAT I'M DOING.



Play/Stop

Code Editor

Open/Save

Ok Let's Make Something

```
point(50,50);

line(40,60, 60, 60);
```

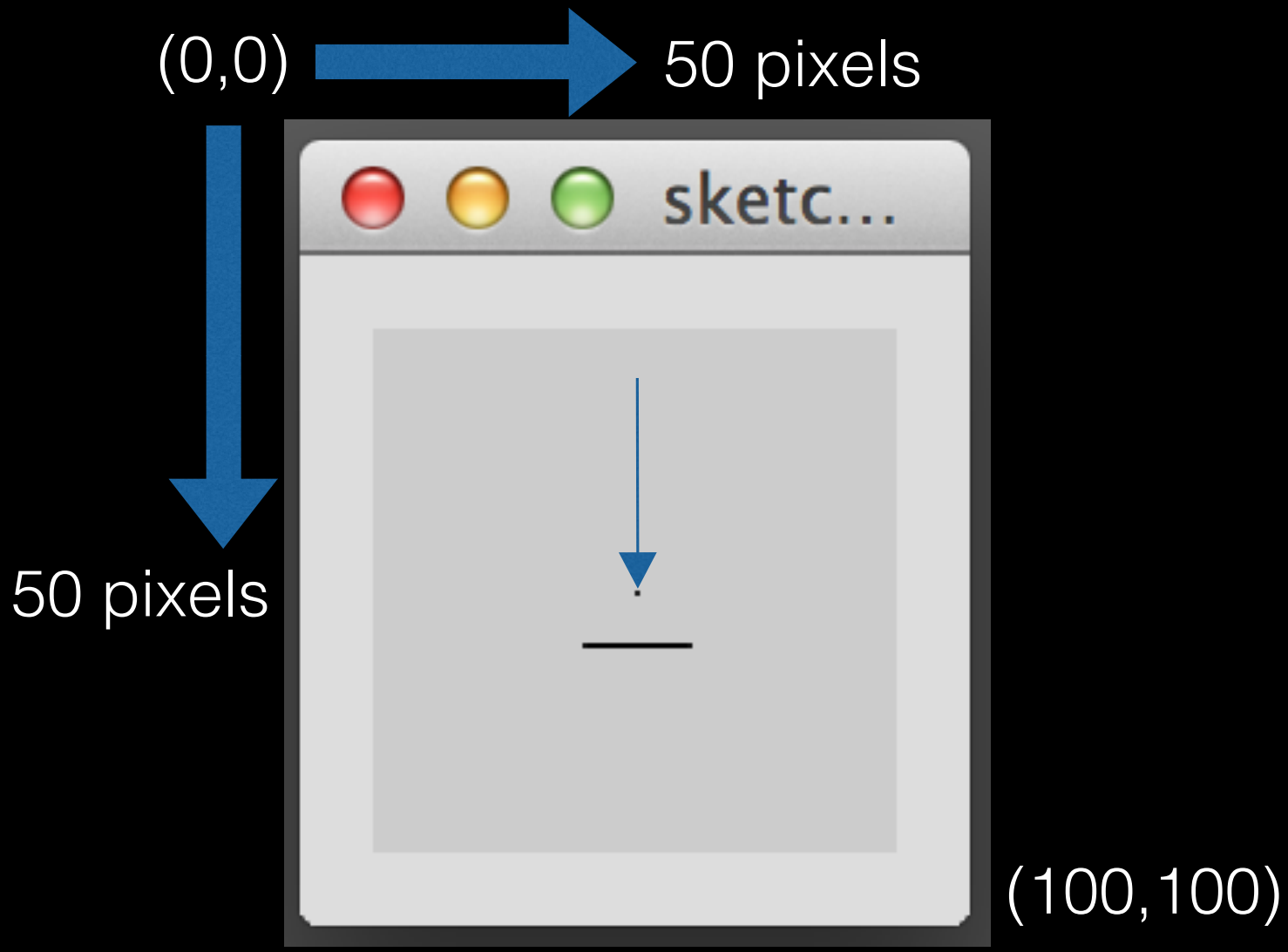

sketch_141011a | Processing 2.2.1
Run
Java
sketch_141011a
point(50,50);
line(40,60, 60, 60);
3
sketc...

The Basics

command(argument1, argument2);

https://processing.org/reference

Coordinates



The Basics

- Commands in Processing contain 3 parts
  - The command name
  - Any arguments for the command (in parenthesis)
  - A semicolon to end the line;
- Note: Each function has their own arguments

The Defaults

- Windows Size: 100 pixels x 100 pixels
  - size(x, y)
- Drawing Color
  - stroke(color)
- Background Color
  - background(color)

Order is Important

- Each command (or function) is executed in order, top to bottom.
- This can be very useful, but also tricky.

Size is Important

```
point(50,50);

size(200,200);
```

—Bad…

tries to draw on the window then change its size

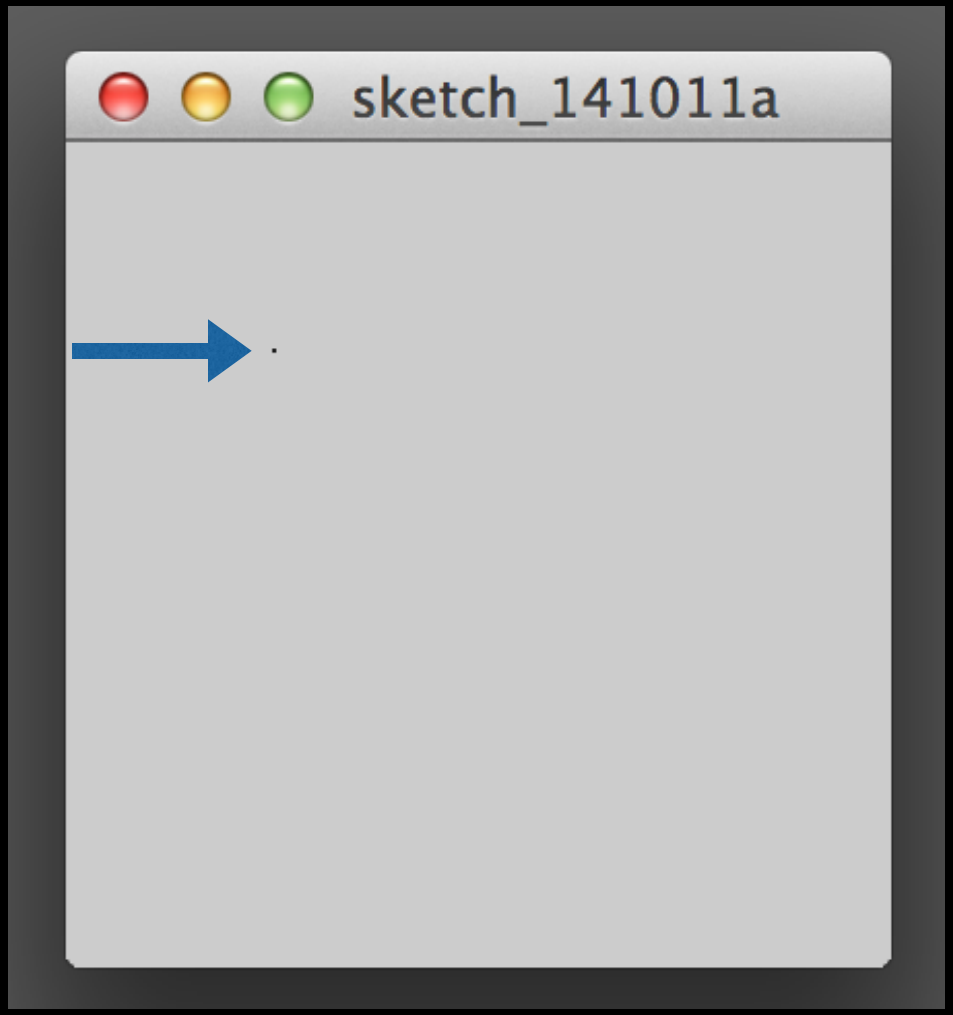
```
size(200,200);

point(50,50);
```

— Good!

sets the size of the window then draws

New Size!

The window is now
200 by 200 pixels



Some Vocabulary

IDE, Editor

- Where all the “programming” takes place
- Where we type code, where we compile code

Window

- When we run our code, Processing gives us a window to “draw” in
- We control the Window from our code

Function

- Functions are the “engines” of processing. They allow you to do things in the Window
- Examples:
  - Change the size of the window
  - Draw in the window

Arguments

- Arguments are used with Functions
- They tell the Function how to operate
- Example:
  - How large to make the window
  - Where to draw

The Function

function(argument1, argument2, etc.);

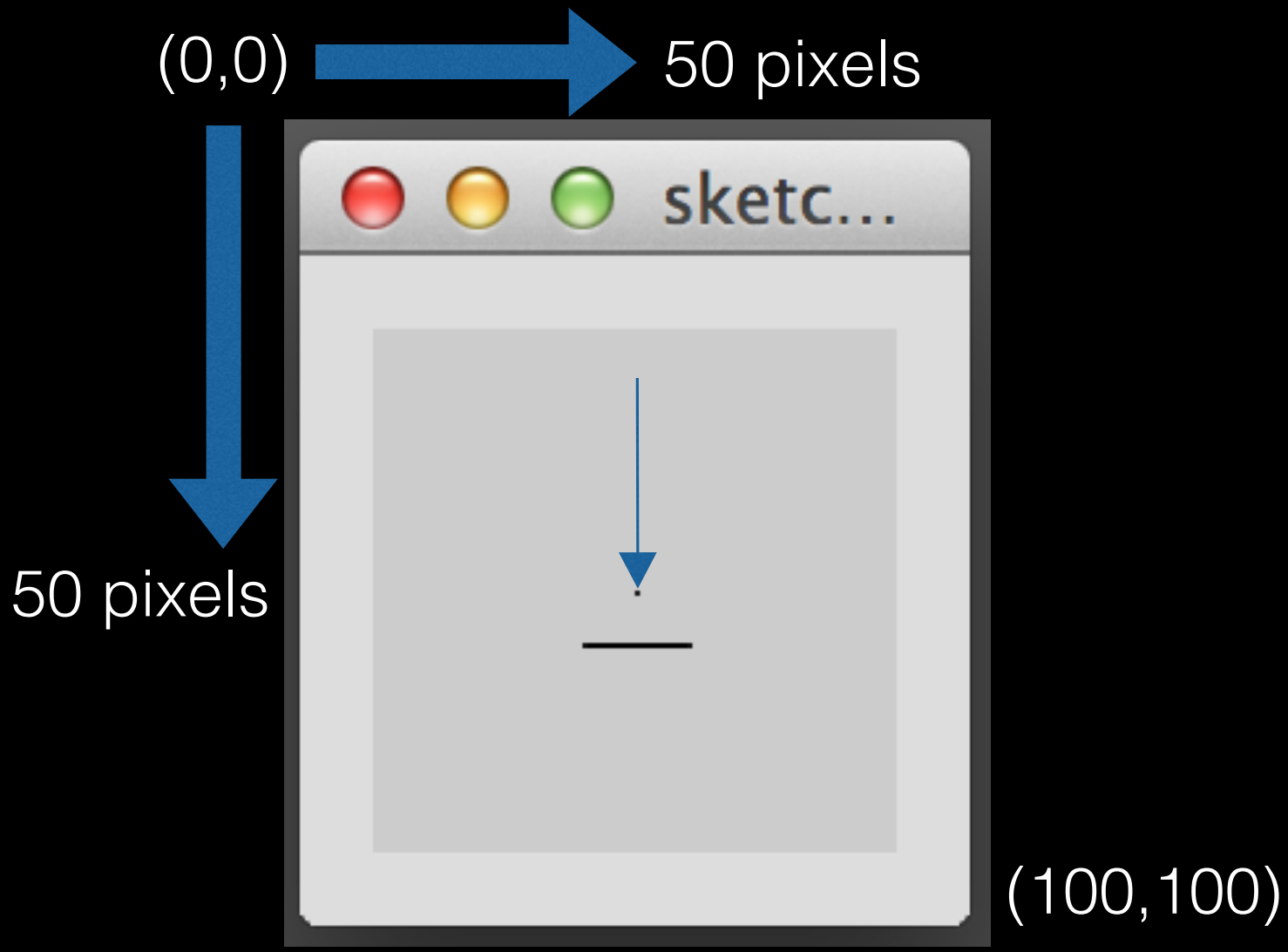
ALL THE FUNCTIONS

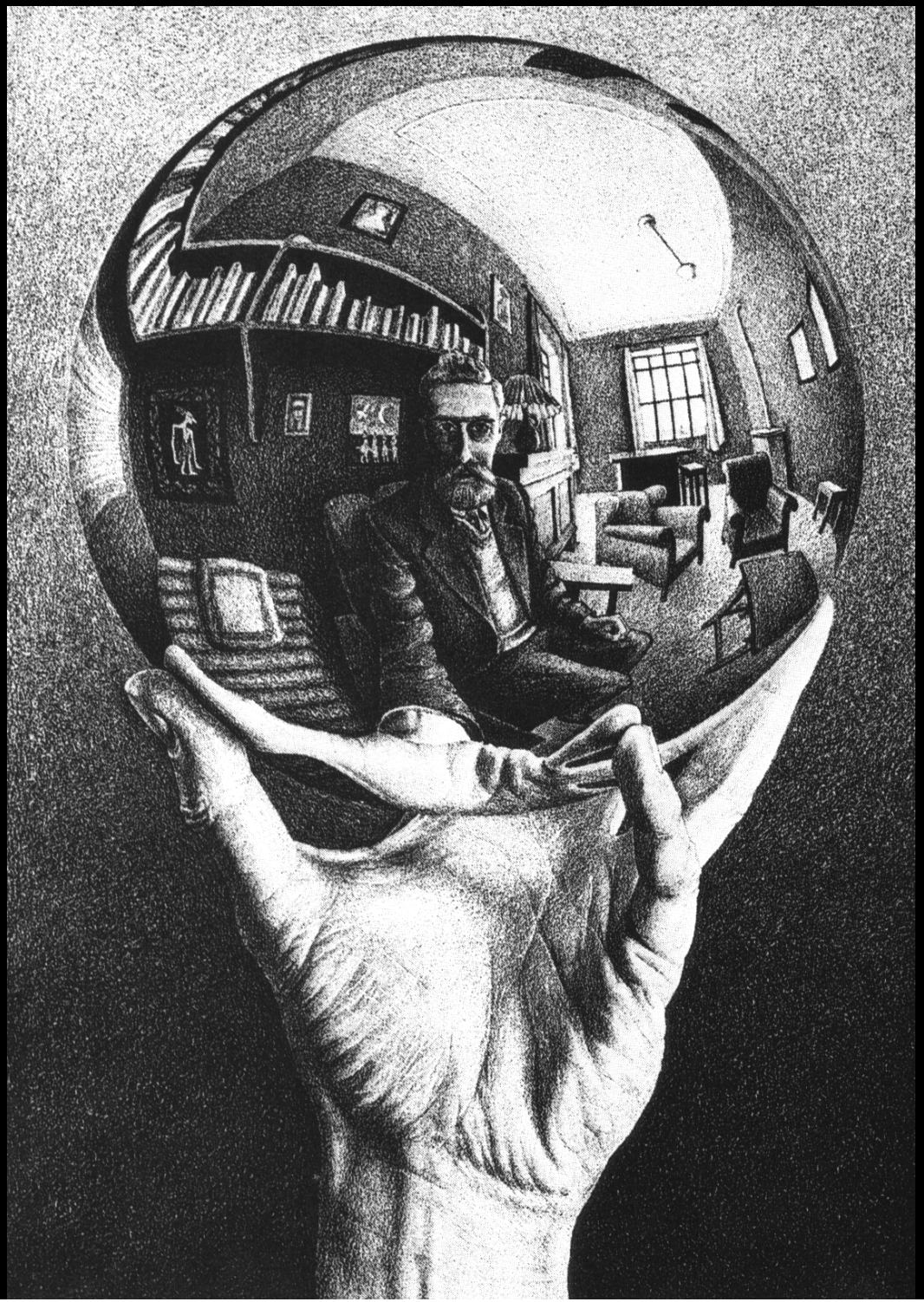
http://www.processing.org/reference/

The Point Function

point(# of pixels over, # of pixels down)

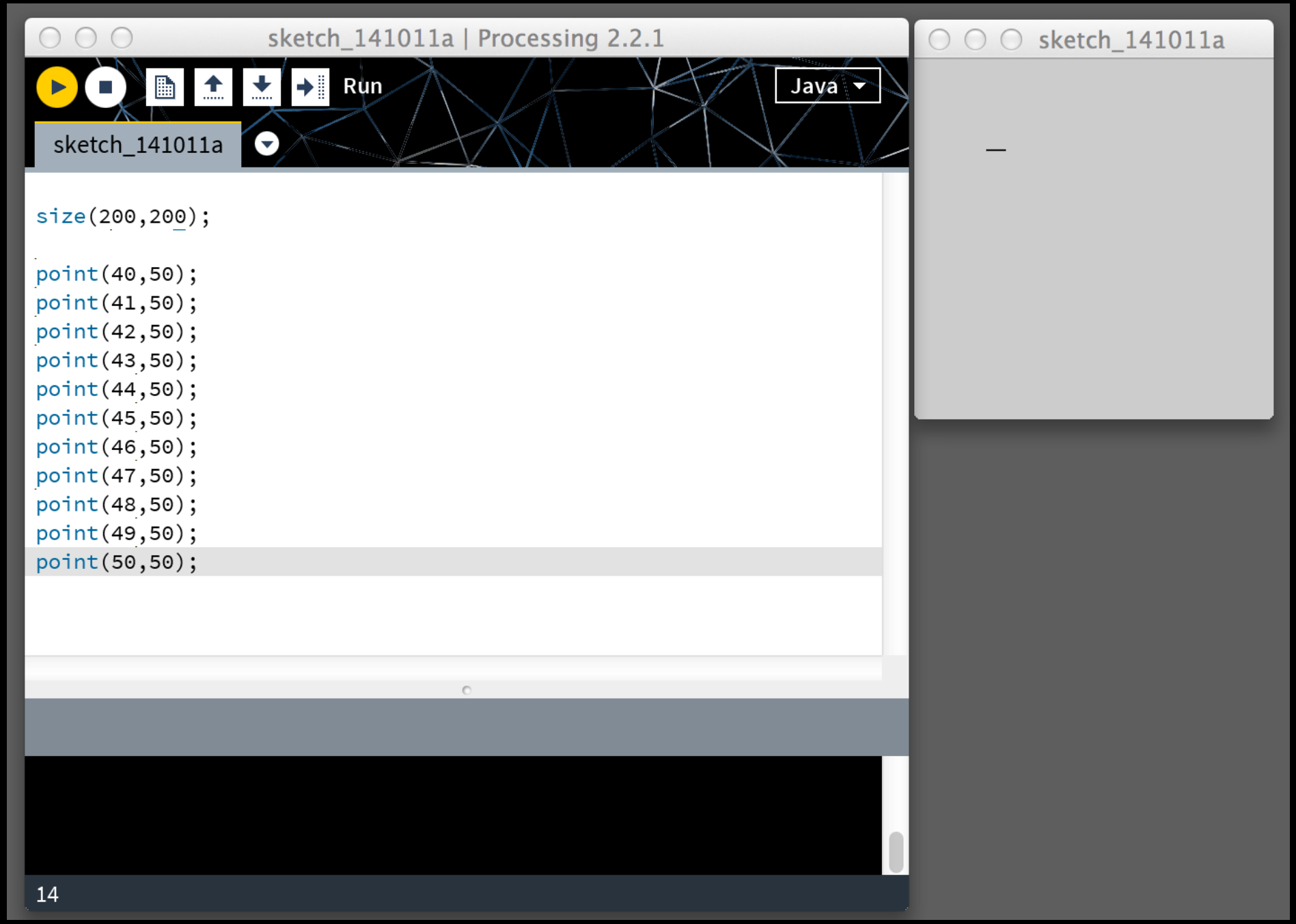
Coordinates





What's in a Line?

- **Code Challenge!!!**
    - In a 200 pixel by 200 pixel window, draw a horizontal line that starts at point(40,50) and extends to the right 10 pixels
- You can only use the size() and point() functions


sketch_141011a | Processing 2.2.1
Run
Java
sketch_141011a
size(200,200);
point(40,50);
point(41,50);
point(42,50);
point(43,50);
point(44,50);
point(45,50);
point(46,50);
point(47,50);
point(48,50);
point(49,50);
point(50,50);
14
sketch_141011a

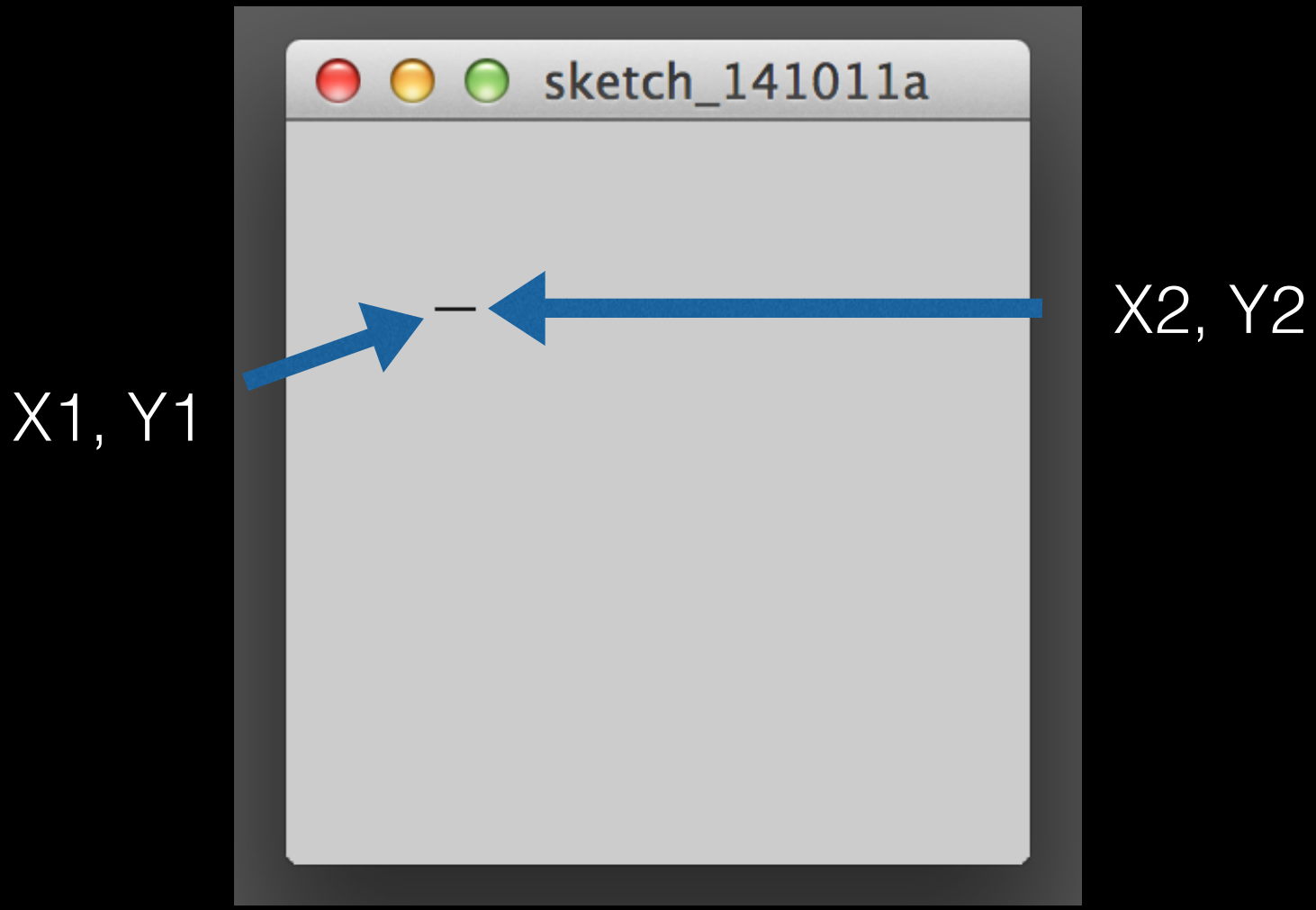
Painting with Points Sucks

- Who would ever want to do that???
- Thankfully, Processing has a number of functions you can use to cheat a little.

Line

line(x1, y1, x2, y2)

Line Coordinates



Code Challenge

- **Let's Make a Grid**
    - In a 100 pixel by 100 pixel window, draw a grid of 6 horizontal lines and 6 vertical lines
        - aka 5 x 5 grid, 20 pixels per square
- You can only use the size(), point(), or line()

What's in a Square?

- **Code Challenge!!!**
    - In a 200 pixel by 200 pixel window, draw a square that starts at point(40,50) and extends to the right 10 pixels and down 10 pixels
- You can use the size(), point(), or line() functions

There's Got to be a Better Way

rect(x,y,width,height)

Color - Stroke and Fill

- Line color is controlled by stroke(color);
  - stroke(255); — White —
  - stroke(0); — Black —
- If you want your line to be in color (not grayscale), use 3 arguments
  - stroke(red, green, blue);

Colored Lines

```
stroke(255,0,0);

line(0,50,100,50);
```

How rect() Works

- rect() draws a bounding box (with stroke()) around the shape you are trying to make.
- Once complete, it fills in the inside of the square

Fill it in with Color

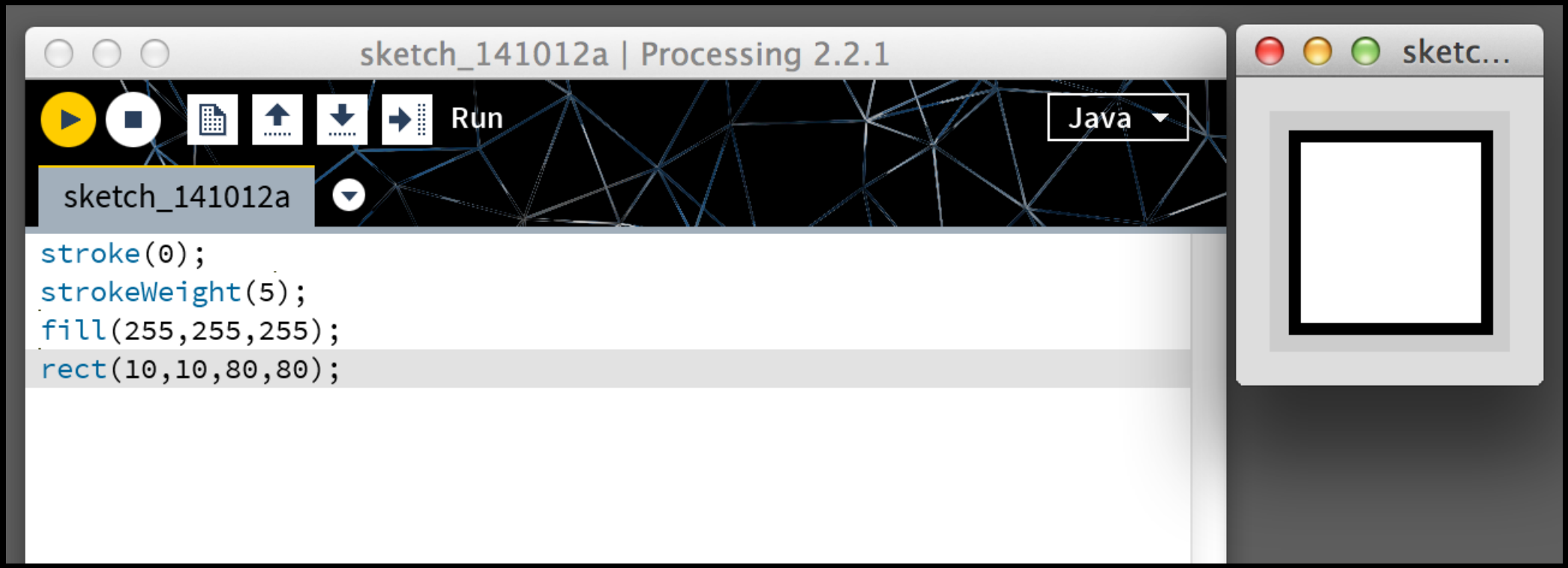
- When making shapes, the color of the outside edge is controlled with stroke()
- The color of the inside of the shape is controlled with fill(). This takes similar arguments to stroke()
    - fill(255,0,0);


sketch_141012a | Processing 2.2.1
Java
sketch_141012a
stroke(255,0,0);
fill(255,0,0);
rect(10,10,80,80);
sketc...

Weigh your Strokes

- The thickness of lines are controlled by strokeWeight(thickness in pixels)
    - strokeWeight(1); — Default, 1 pixel
    - strokeWeight(10); — 10 pixels
- The thickness of the line extends in two directions!!


sketch_141012a | Processing 2.2.1
Run
Java
sketch_141012a
stroke(0);
strokeWeight(15);
line(0,50,100,50);
sketc...


sketch_141012a | Processing 2.2.1
Run
Java
sketch_141012a
stroke(0);
strokeWeight(5);
fill(255,255,255);
rect(10,10,80,80);
sketc...

Some Gotchas…

- strokeWeight() adds to the overall size of your shape
- The shape always starts and ends at the pixels you specify, and extends the distance of the strokeWeight() on either side of the line being drawn

Drawing Order

- What happens when you want to draw more than one shape, line, etc on the screen?
- Processing always draws your commands in order, if there's any overlap, Processing will draw over what is already on the screen

Drawing Order

- This applies to stroke(), fill(), strokeWeight() etc as well
- Imagine you are drawing with colored pencils. commands like stroke() and fill() simply change the pencil that you are using when drawing.

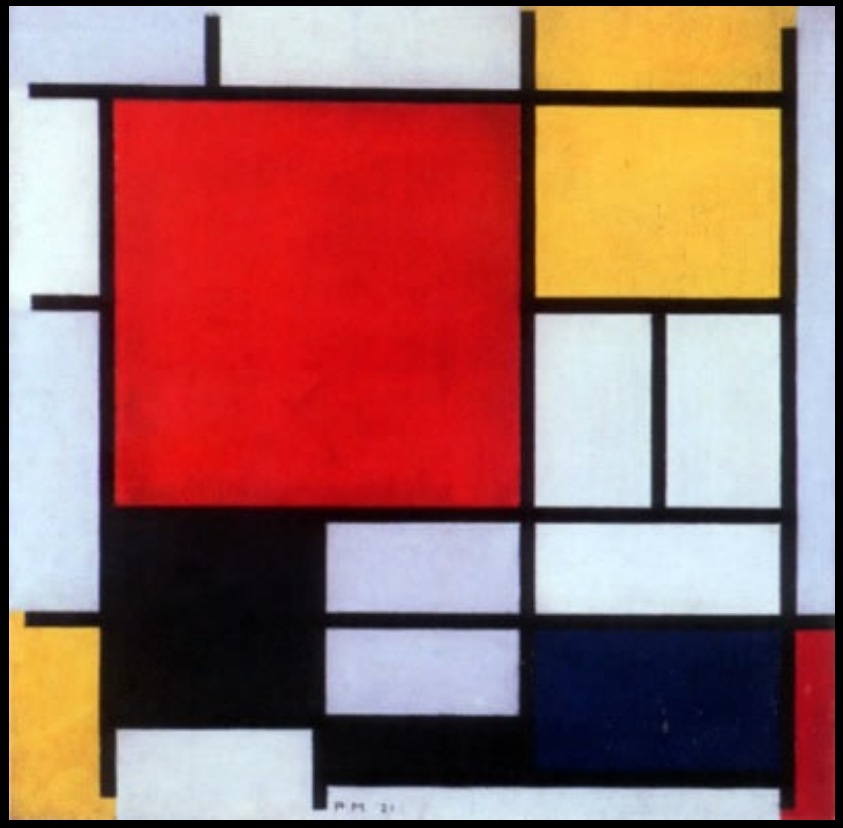

sketch_141012a | Processing 2.2.1
Java
sketch_141012a
stroke(0);
strokeWeight(5);
fill(255,255,255);
rect(10,10,60,60);
stroke(255);
strokeWeight(2);
fill(255,0,0);
rect(30,30,60,60);
sketc...


ONE ART, PLEASE!
quickmeme.com

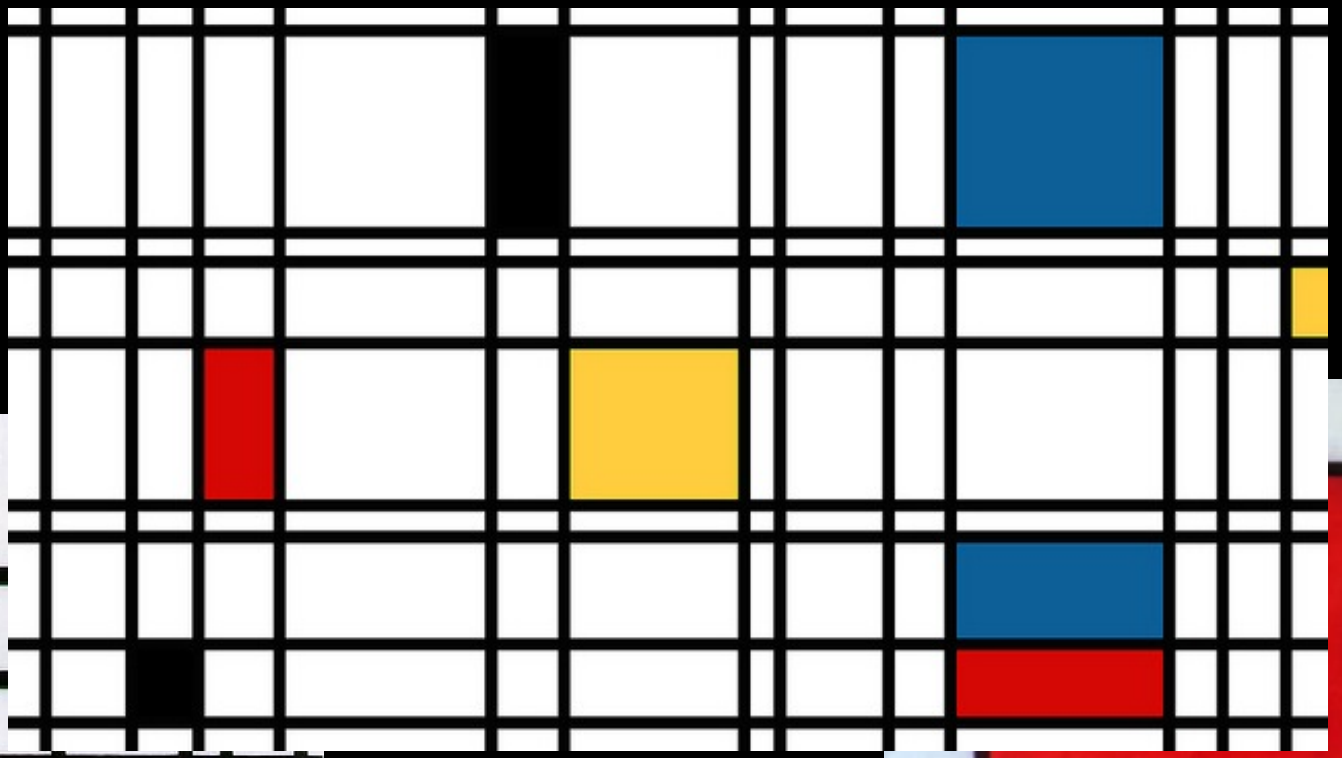
SUPER AWESOME CODE CHALLENGE

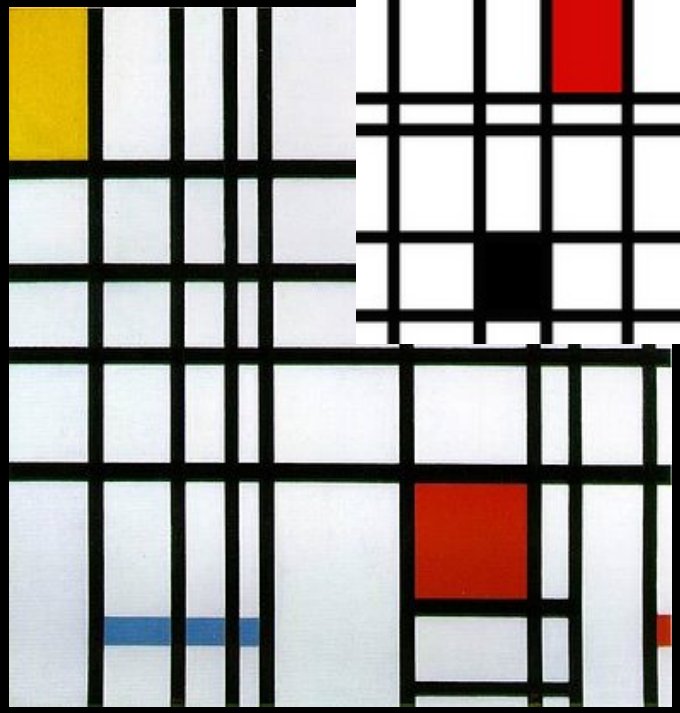
Make a Mondrian!

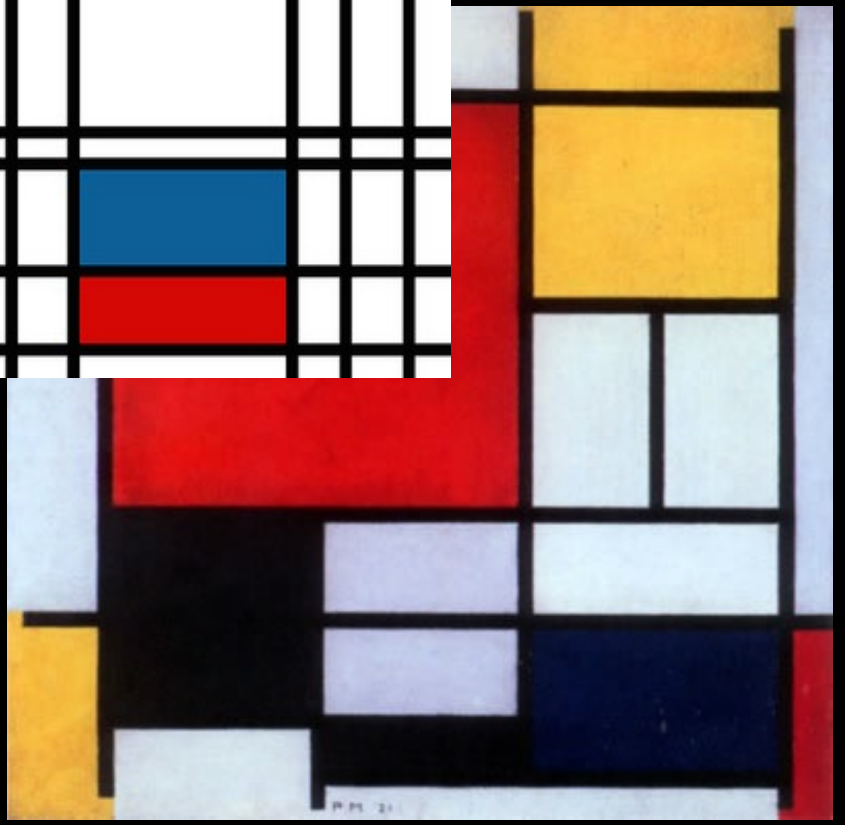
Piet Mondrian



Mondrian - The Basics

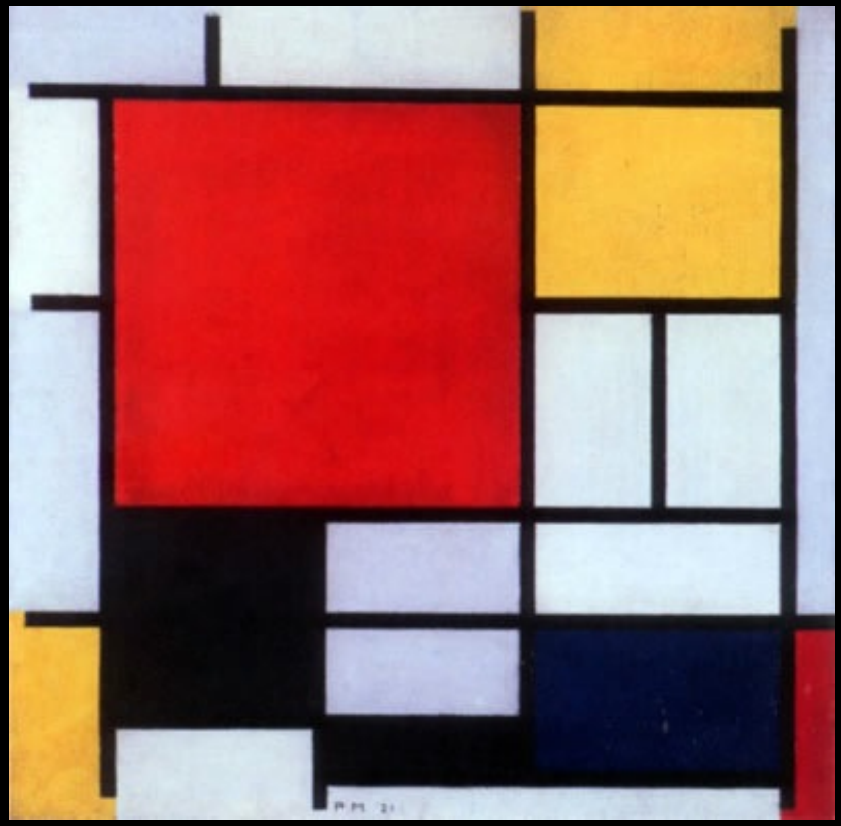




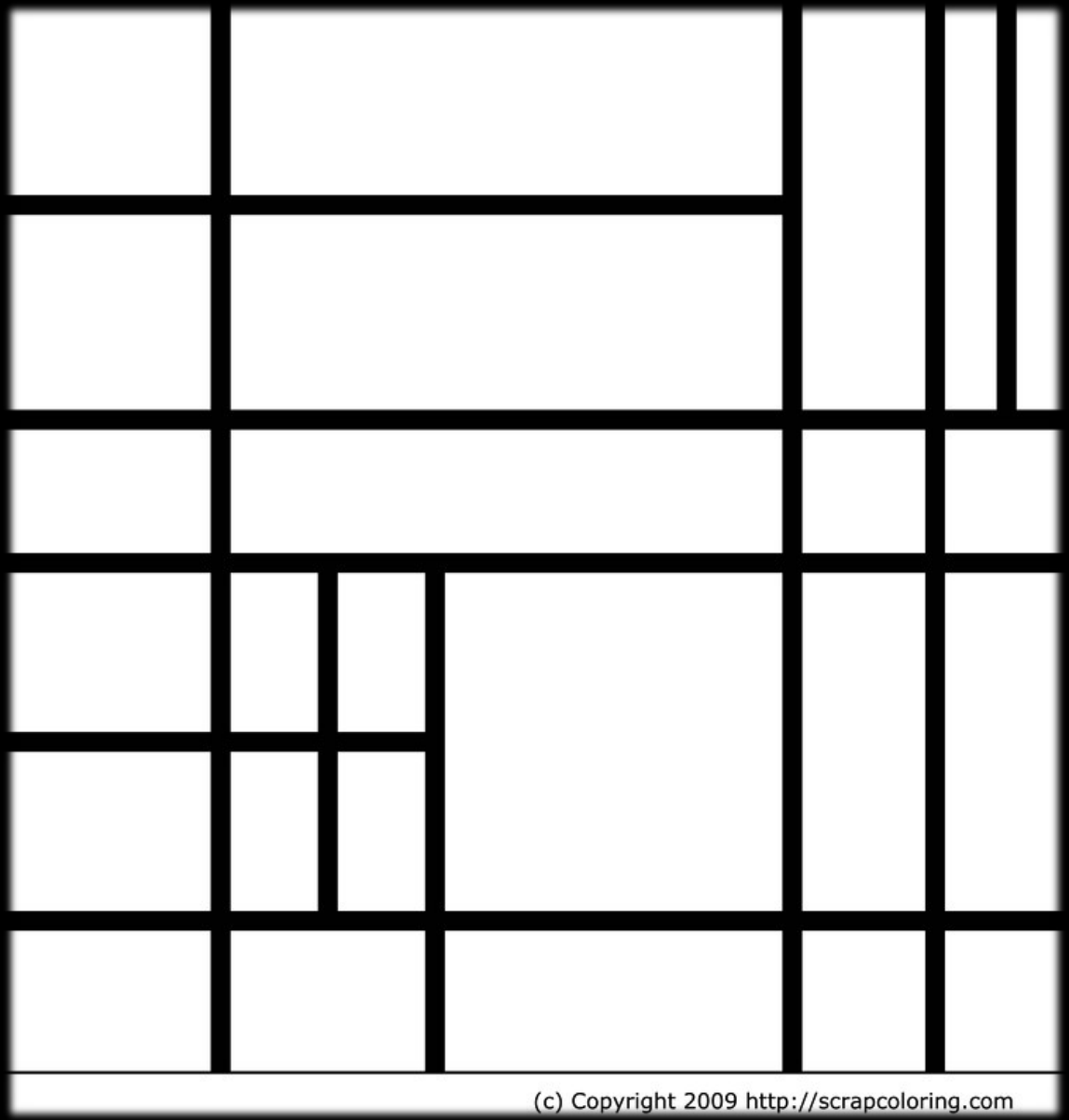


The Rules

- At least 6 horizontal lines and 6 vertical lines
- The screen must be 500x500 or greater
- The lines cannot be all equidistant (i.e. don't draw a grid)
- Rectangles and lines must match up
- Rectangles cannot be all the same size
- Rectangles must be at least 3 different colors



Mondrian - Cheat Sheet



Fín