Programming Fundamentals

An Introduction to the module and Processing

Produced by: Dr. Siobhán Drohan
Mr. Colm Dunphy
Mr. Diarmuid O' Connor

Waterford Institute *of* Technology
INSTITIÚID TEICNEOLAÍOCHTA PHORT LÁIRGE

Department of Computing and Mathematics
http://www.wit.ie/

Agenda

- *Module Structure / Approach:*
  - *Introducing your lecturers*
  - *Structure of the module*
  - *Troubleshooting labs*
  - *Module assessment*
  - *Ethos*

- *Introduction to Processing and the PDE.*

- *Starting to Code in Processing.*

Introducing your lecturers

Colm Dunphy

- Profile:https://www.wit.ie/about_wit/contact_us/staff_directory/colm_dunphy
- Email:cdunphy@wit.ie

Diarmuid O'Connor

- Profile:https://www.wit.ie/about_wit/contact_us/staff_directory/diarmuid-oconnor
- Email: doconnor@wit.ie

Structure of the module

12 weeks of delivery

Lectures ?sdr?

Labs ?sdr?

Mon 12 – 1:30pm

Thurs 12 – 1:30pm

Tues 10am-12

Fri (A) 10am-12

Fri (B) 12-2pm

Fri (C) 2-4pm

Structure of the module

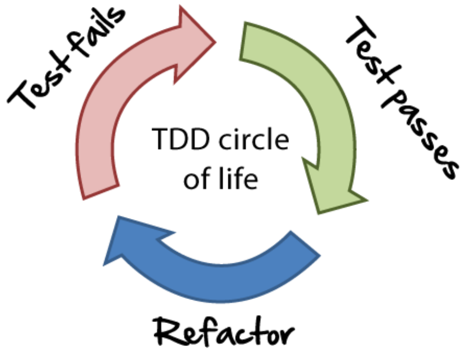
moodle

Adobe® Connect™

GitHub

gitter

Structure of the module



Week Starting…	Topic	IDE	Assessment (100% CA)
Week 1 (22nd Jan)	Static and Animated Drawings, Sequence, Data Types	Processing	
Week 2 (29th Jan)	Selection (if), Iteration (loops), Events	Processing	*Assign 1 spec released*
Week 3 (5th Feb)	Using and Writing Methods	Processing	
Week 4 (12th Feb)	Strings, Classes, Objects	Processing	
MIDTERM (19th Feb)	MIDTERM	MIDTERM	
Week 5 (26th Feb)	Primitive Arrays and More on Classes	Processing	
Week 6 (5th March)	Building the Game of Pong (released end of week 4)	Processing	*Assignment 1 due Sunday*
Week 7 (12th March)	IntelliJ, Basic I/O, Array Recap, Collections (ArrayList)	IntelliJ	*Assign 2 spec released*
Week 8 (19th March)	Collections (ArrayList), Menu Driven Apps, Persistence	IntelliJ	
EASTER (26th March)	EASTER HOLIDAYS	EASTER	
EASTER (2nd April)	EASTER HOLIDAYS	EASTER	*Assign 2 due Sunday*
Week 9 (9th April)	XML, Exceptions, Collections (Maps, Sets)	IntelliJ	*Assign 3 spec released*
Week 10 (16th April)	Inheritance, Polymorphism, Abstraction	IntelliJ	
Week 11 (23rd April)	TDD and JUnit	IntelliJ	
Week 12 (30th April)	Interfaces	IntelliJ	*Assign 3 due Sun 20th May*

Assignment structure

- 100% Continuous Assessment (CA).
- 3 assignments:
  - Assignment 1 (30%) – due Sunday 10th March, 5PM
  - Assignment 2 (20%) – due Sunday 8th April, 5PM
  - Assignment 3 (50%) – due Sunday 20th May, 5PM
- Hard deadlines; extensions only permitted if mitigating circumstances apply.
- Individual assignments (no team-based ones).
- Submit via Moodle assignment dropboxes.

Troubleshooting labs ...during the two hour session

Post the issue in Gitter; think of it as asking a question in a traditional classroom. Include any screen shots, screen recordings, etc you think might help solve the problem.

We encourage classmates to help each other, so if you know the answer to another student's issue, please do respond.

All our responses will be via Gitter so that all students can see the resolution.

Note: for private issues, chat is also possible with us privately in Gitter (or email).

Troubleshooting labs …outside of two hour session

Search Gitter Chatroom

Search Google / StackOverflow (or equivalent)

Post the issue in Gitter Chatroom

Ethos

- Self-directed learning outside of lectures / labs.
- Inquisitive and motivated.
- Helpful to peers.
- Engagement and staying current with the module.
- All work submitted must be your own work.
    - Note: all code/approaches given in the module by us can be re-used / re-purposed in your assignments.

Introduction to Processing

What is Processing?

"Processing is a programming language, development environment, and online community."

Source: https://processing.org/

- Some online examples developed using Processing:
  - http://www.thesheepmarket.com/
  - http://balldroppings.com/js/
  - http://www.openprocessing.org/browse/

What is Processing?

Processing…

…can be used to develop static or interactive online material and data visualisations.

…is often used by visual artists.

…produces visual and interactive representations of programming code.

What is Processing?

- Different programming languages can be used with Processing e.g. :
  - Java: we will use this language.
  - JavaScript
  - Python
  - CoffeeScript
  - Etc.

Why are we using Processing?

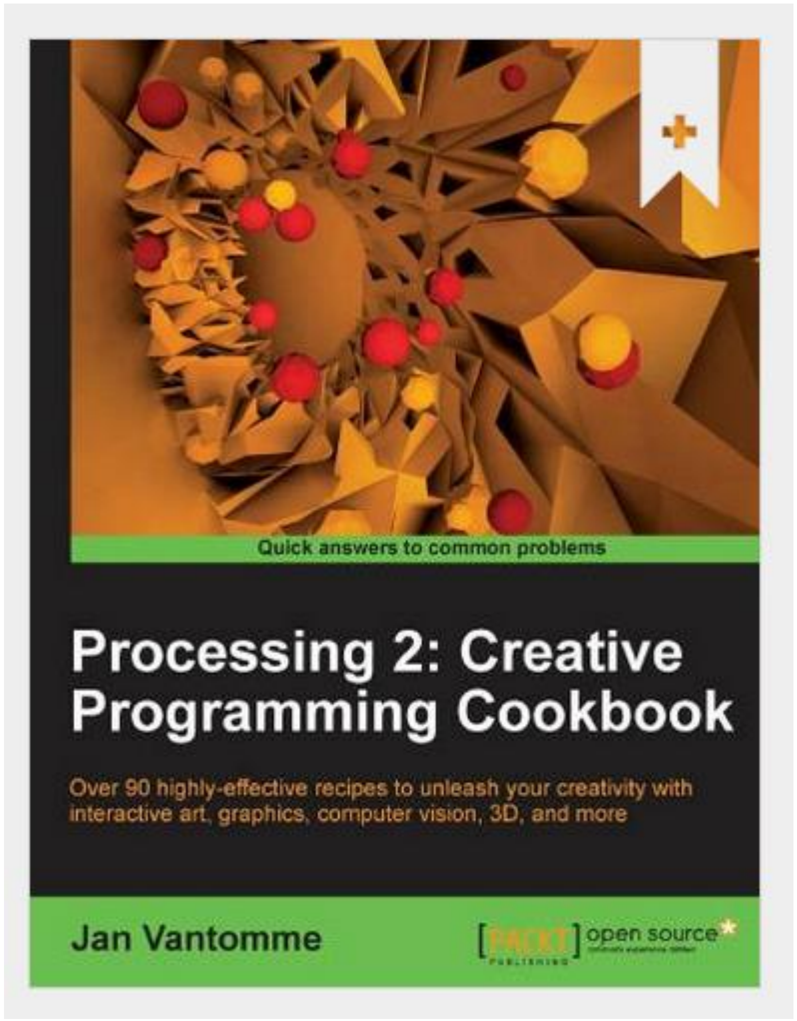
Processing is increasingly used

to teach computer

programming fundamentals

(https://processing.org/overview/)

Some eBooks in WIT library



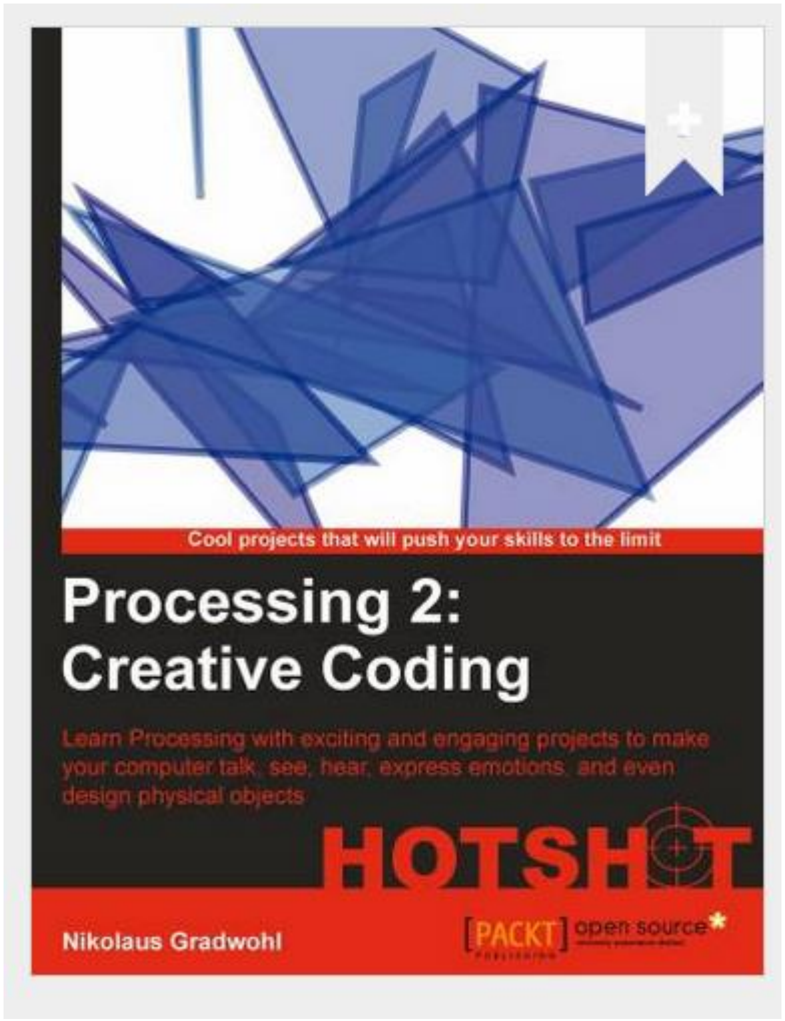
Quick answers to common problems

Processing 2: Creative Programming Cookbook

Over 90 highly-effective recipes to unleash your creativity with interactive art, graphics, computer vision, 3D, and more

Jan Vantomme

[PACKT] open source*



Cool projects that will push your skills to the limit

Processing 2: Creative Coding

Learn Processing with exciting and engaging projects to make your computer talk, see, hear, express emotions, and even design physical objects

HOTSHOT

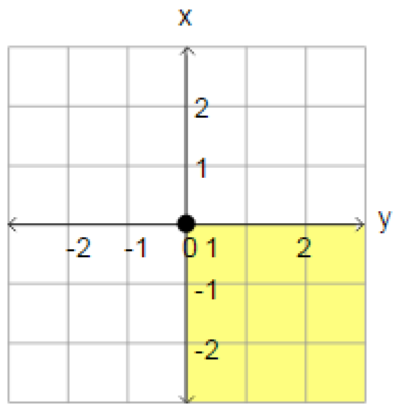
Nikolaus Gradwohl

[PACKT] open source*

Starting to Code in Processing

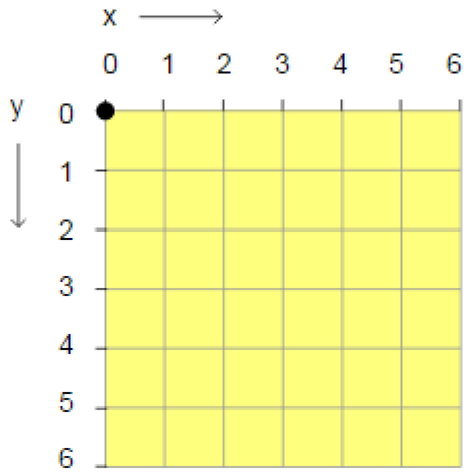
Coordinate System in Computing

In Geometry,
we use this type of
coordinate system:



point (0,0) is in the centre.

In Computing, we use this type of coordinate system to represent the screen:

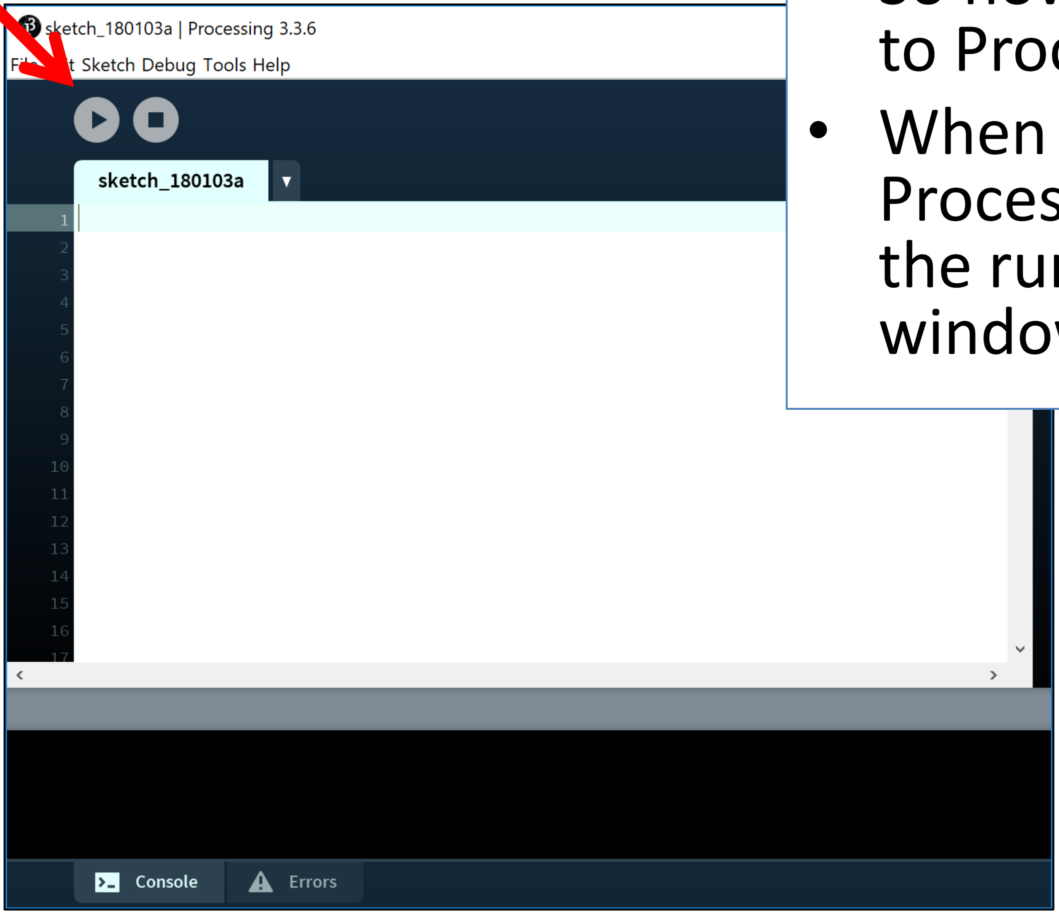


point (0,0) is in the top left hand corner. Each number is a pixel.

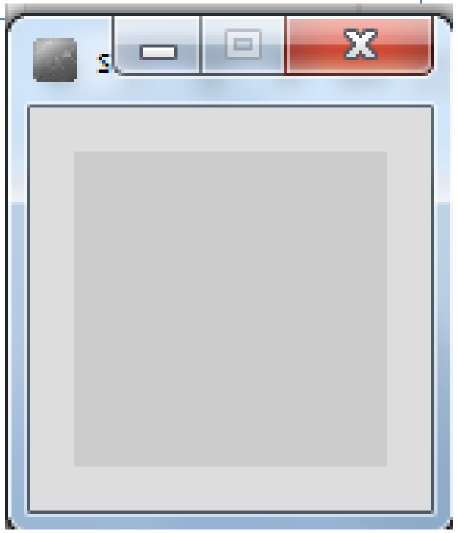
https://processing.org/

Coordinate System in Computing

Run button


sketch_180103a | Processing 3.3.6
File Edit Sketch Debug Tools Help
sketch_180103a
Console
Errors

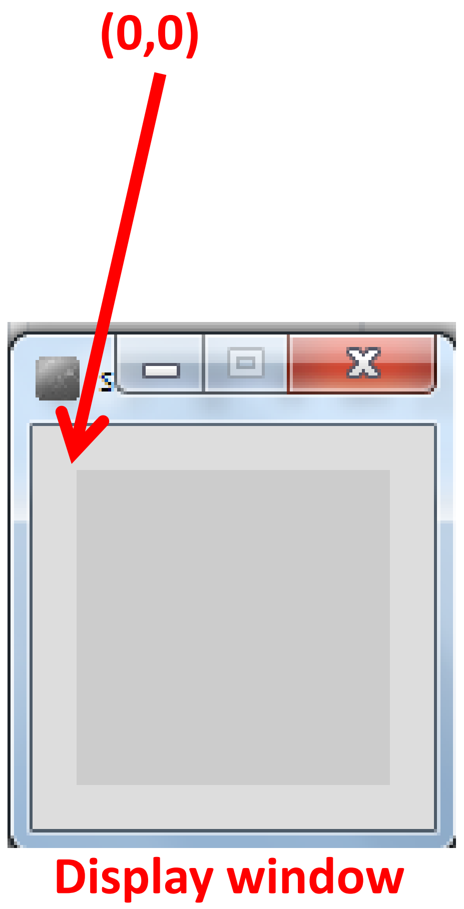
- So how does this relate to Processing?
- When you open Processing and click on the run button, a display window pops up.



Display window

Coordinate System in Computing

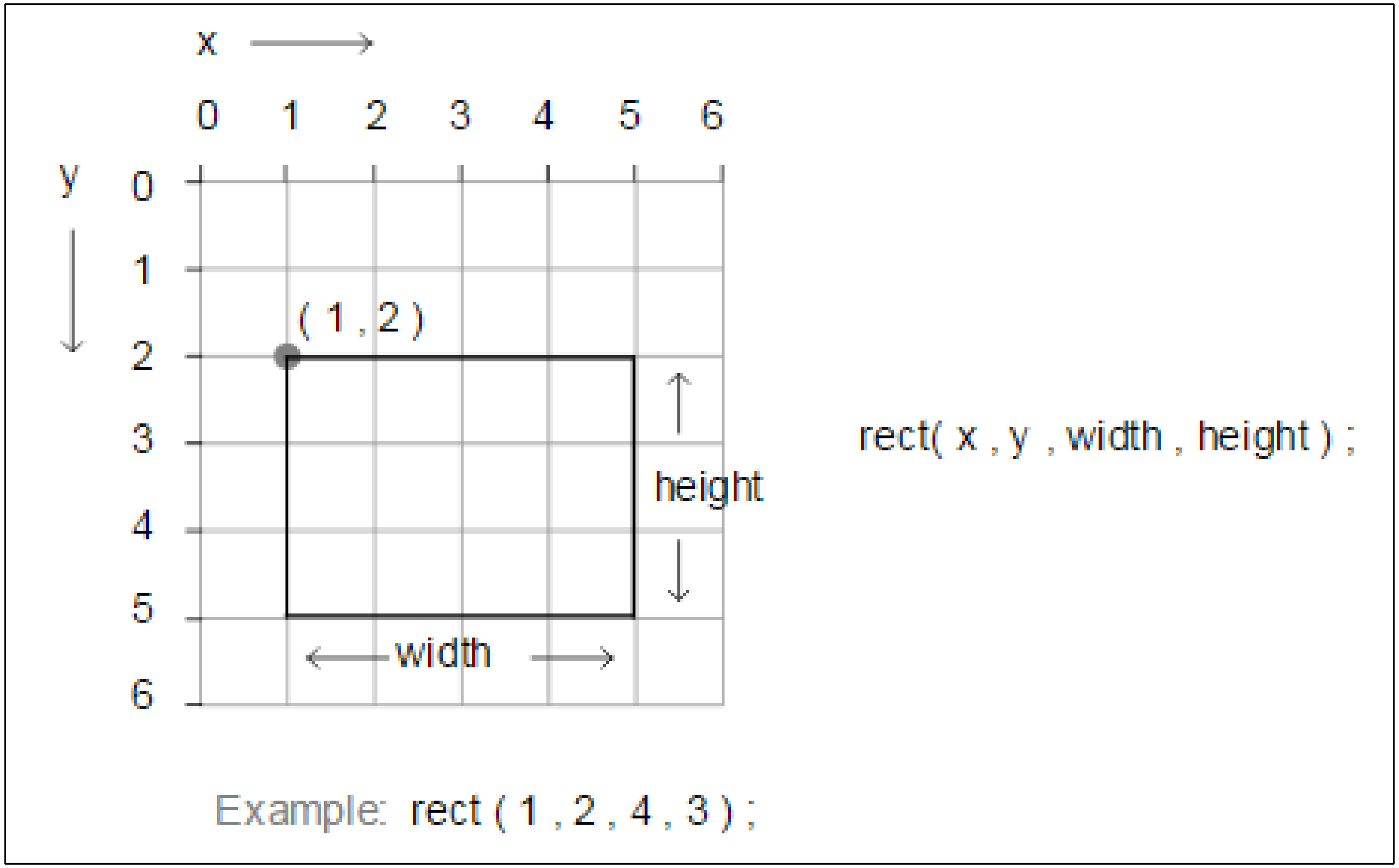
- The display window is where your code is run/ displayed.
- It follows the rules of the Computing coordinate system i.e. the top left hand corner is (0,0).
- A point (10,20) is 10 pixels to the right of (0,0) and 20 pixels below (0,0).



Functions in Processing

- Processing comes with several pre-written functions that we can use.
- A function comprises a set of instructions that performs some task.
- When you call the function, it performs the task.
- We will now look at functions that draw the following shapes:
  - Rectangle, square, line, oval and circle.

rect()

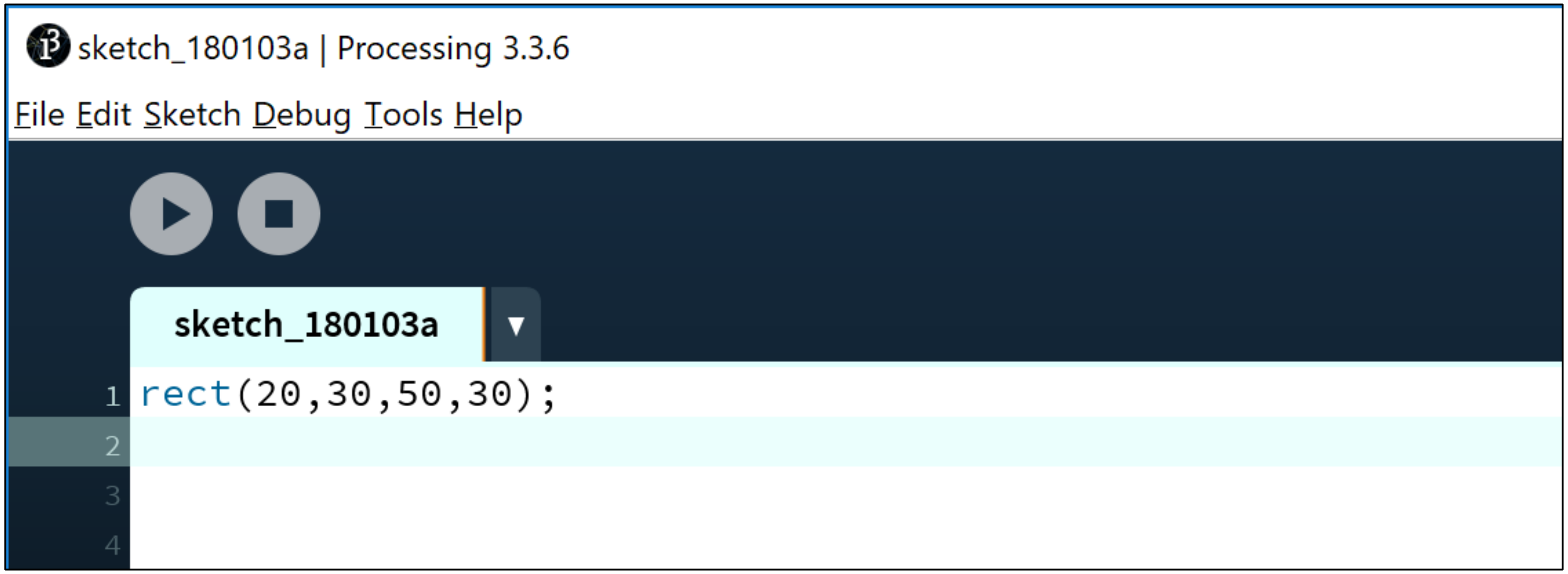


```
rect( x , y , width , height ) ;
```

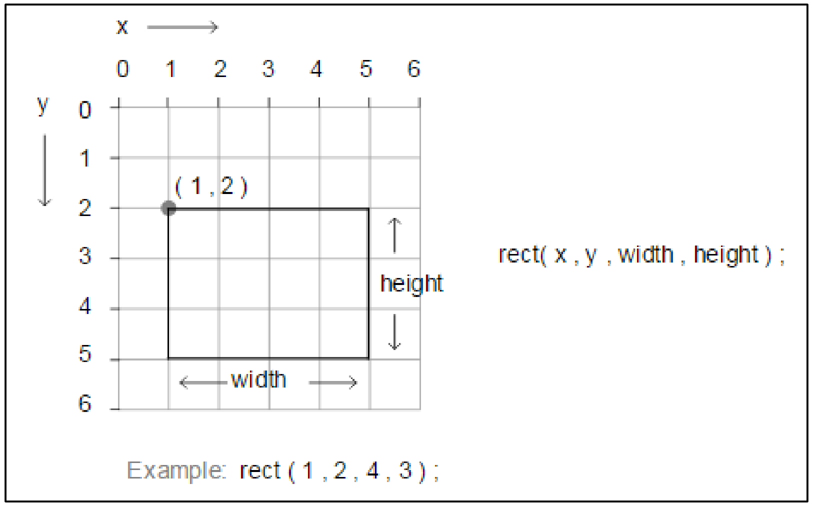
```
Example:  rect ( 1 , 2 , 4 , 3 ) ;
```

https://processing.org/

rect() – drawing a rectangle


sketch_180103a | Processing 3.3.6
File Edit Sketch Debug Tools Help
sketch_180103a

```
rect(20,30,50,30);
```

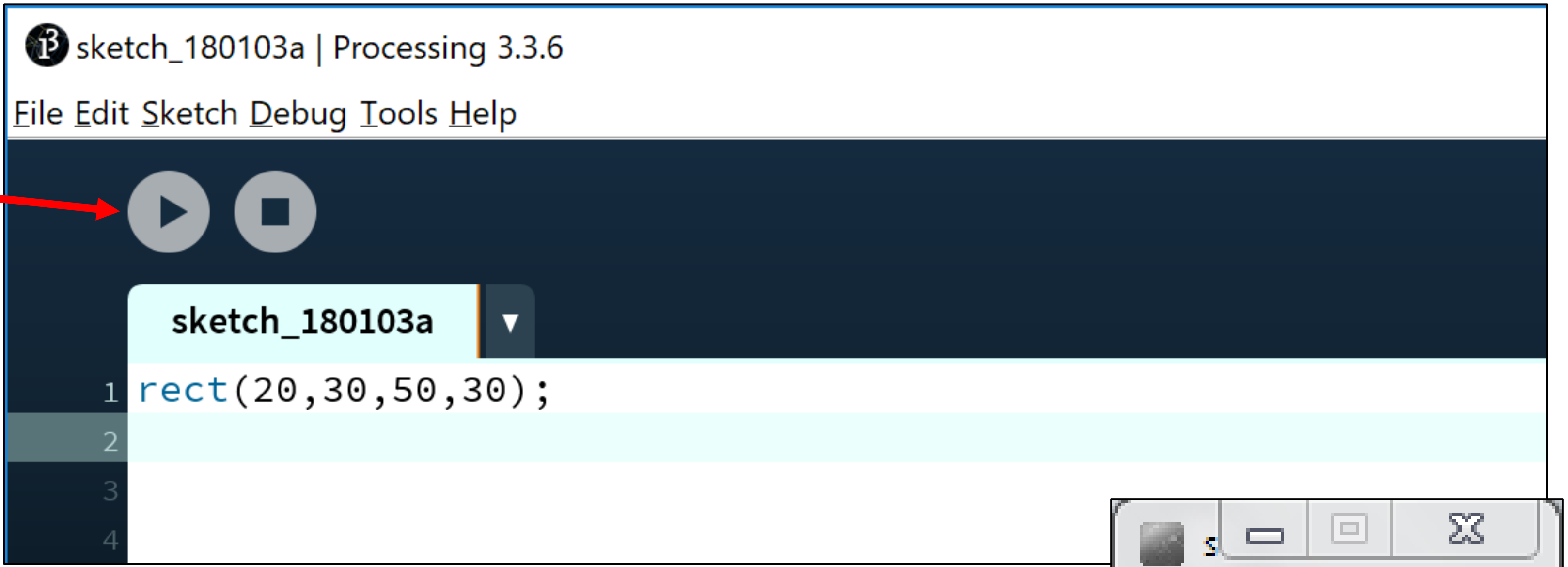


rect(x , y , width , height) ;

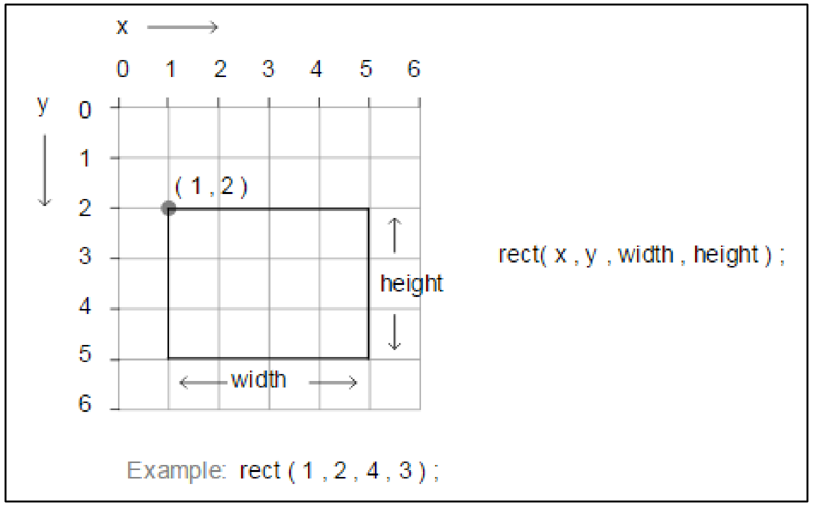
Example: rect (1 , 2 , 4 , 3) ;

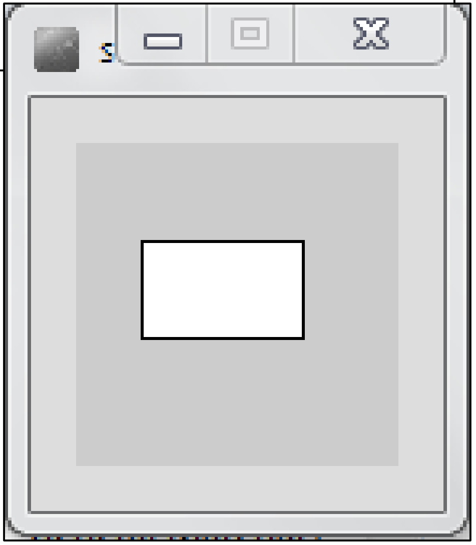
rect() – drawing a rectangle

Click to Run

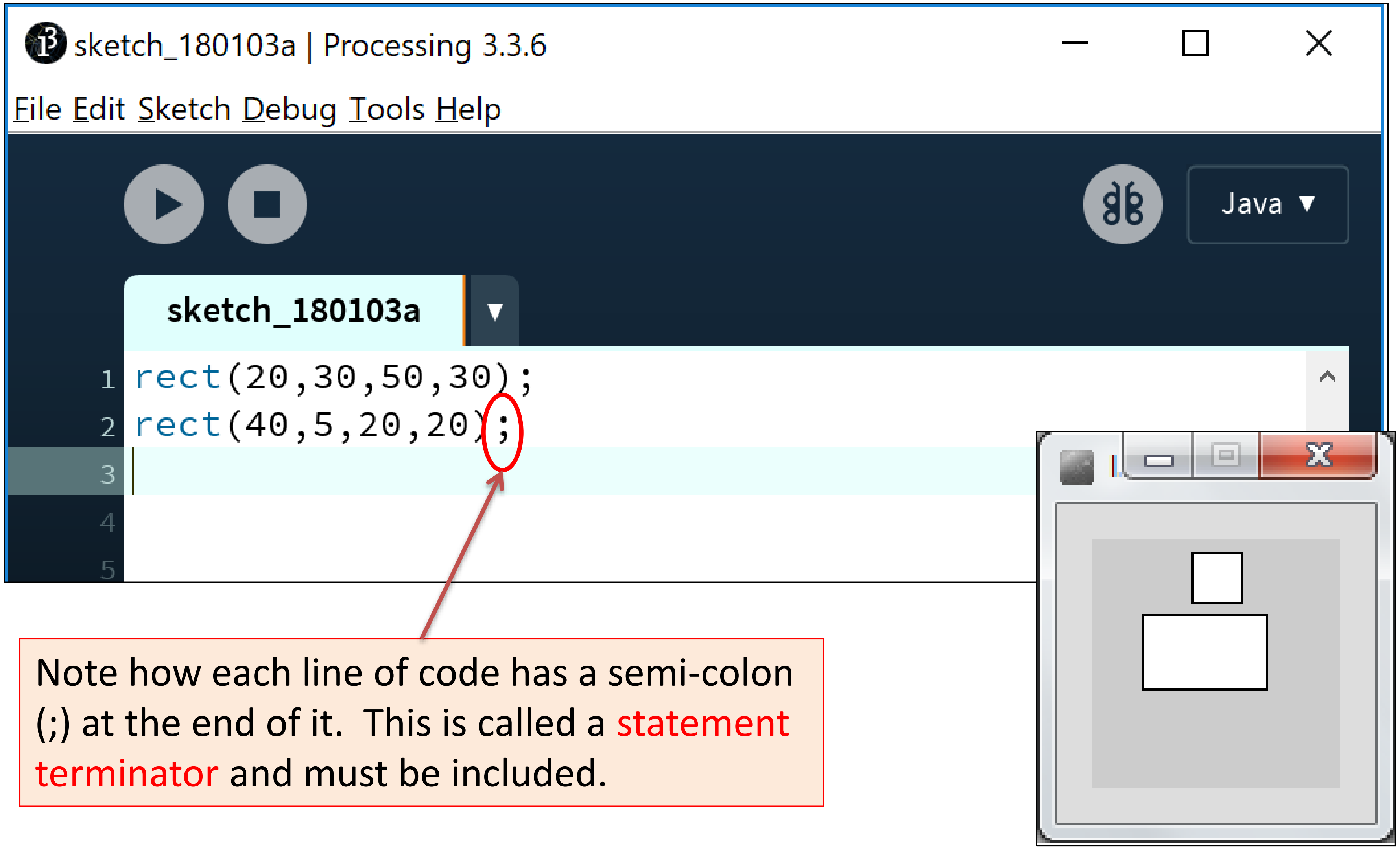

sketch_180103a | Processing 3.3.6
File Edit Sketch Debug Tools Help
sketch_180103a

```
rect(20,30,50,30);
```


x
y
0 1 2 3 4 5 6
(1,2)
height
width
rect(x , y , width , height) ;
Example: rect (1 , 2 , 4 , 3) ;



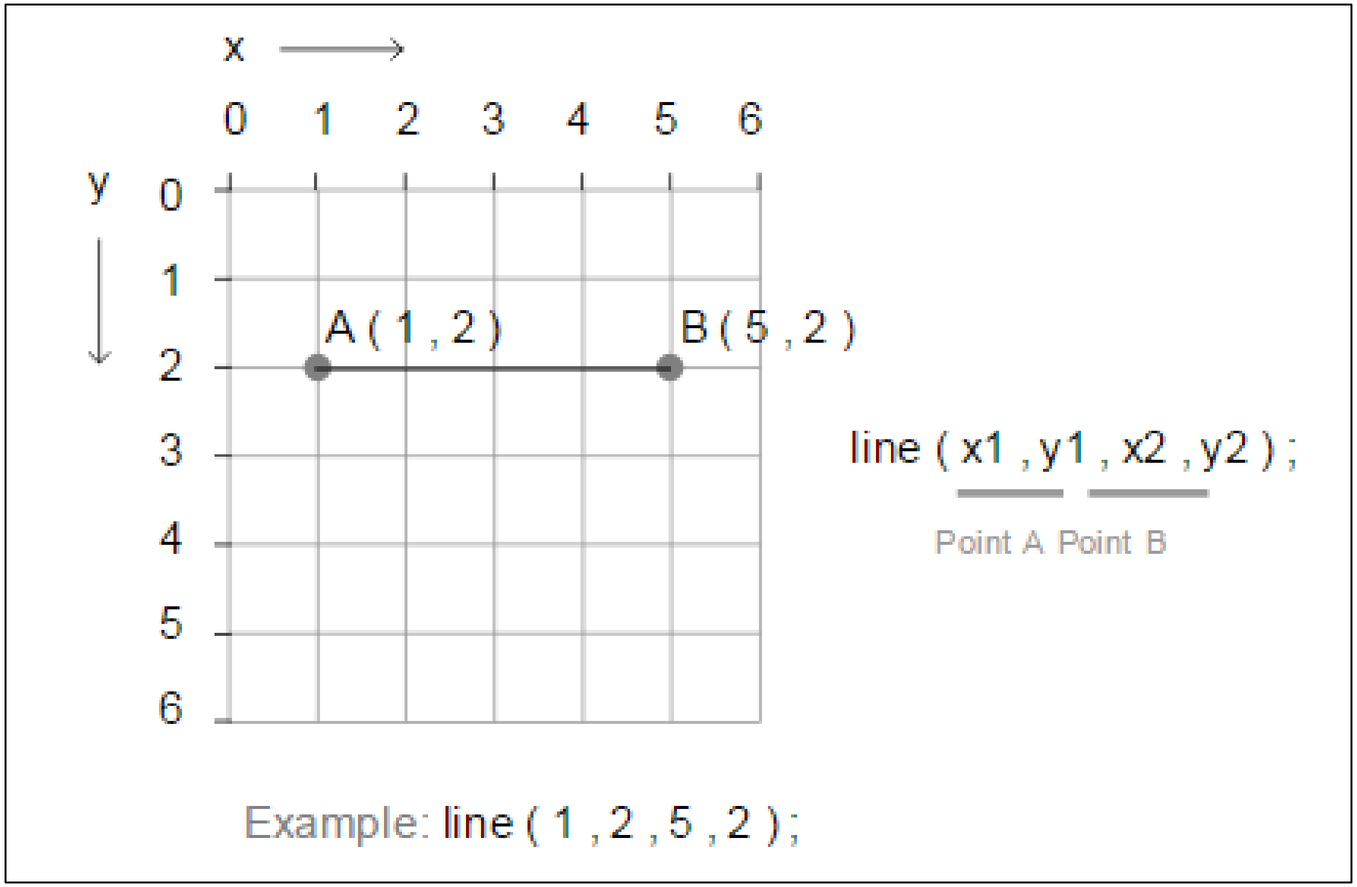
rect() – drawing a square


sketch_180103a | Processing 3.3.6
File Edit Sketch Debug Tools Help
Java ▼
sketch_180103a

```
rect(20,30,50,30);
rect(40,5,20,20);
```

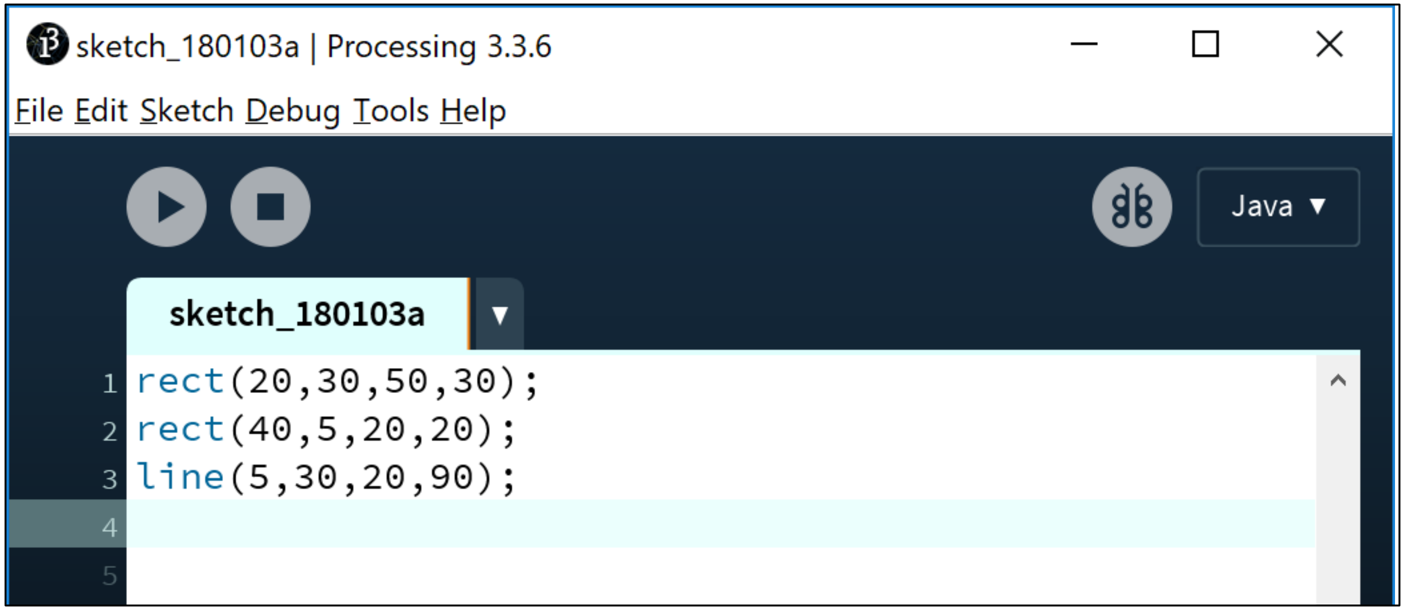
Note how each line of code has a semi-colon (;) at the end of it. This is called a statement terminator and must be included.

line()

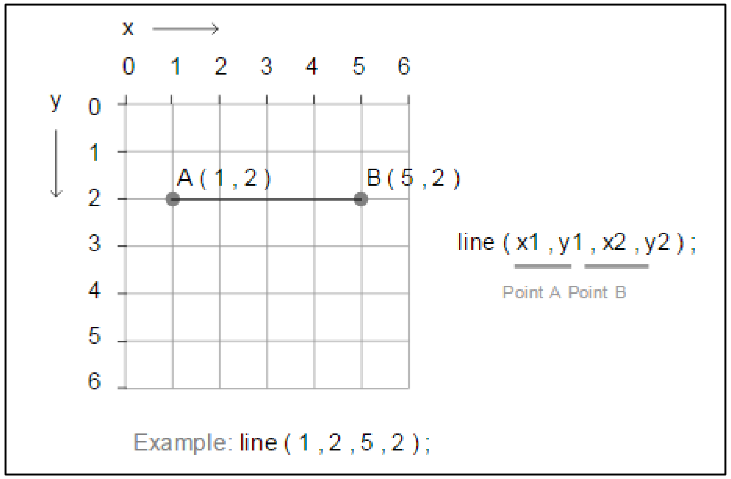


https://processing.org/

line () – drawing a line

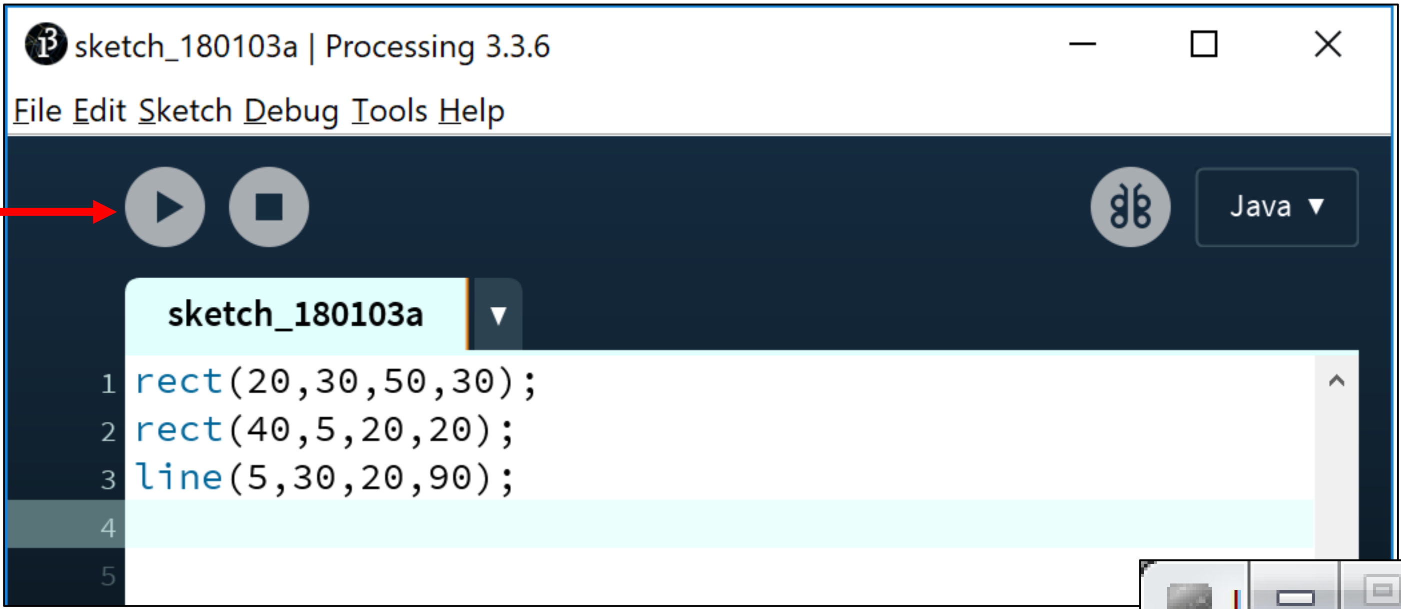

sketch_180103a | Processing 3.3.6
File Edit Sketch Debug Tools Help
Java
sketch_180103a

```
rect(20,30,50,30);
rect(40,5,20,20);
line(5,30,20,90);
```

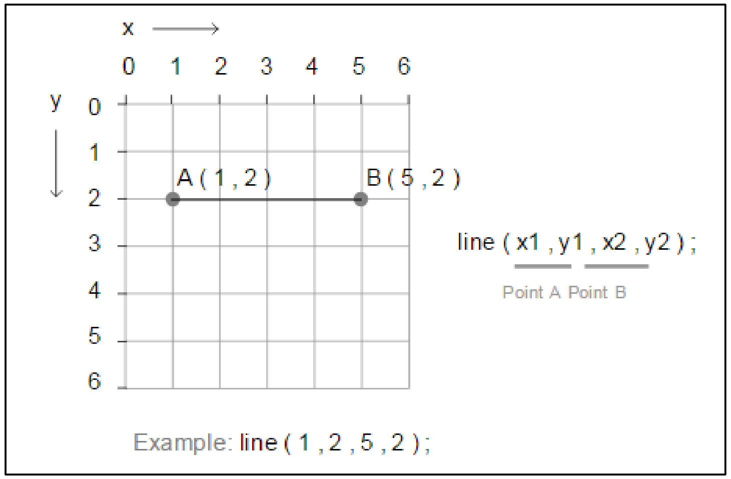


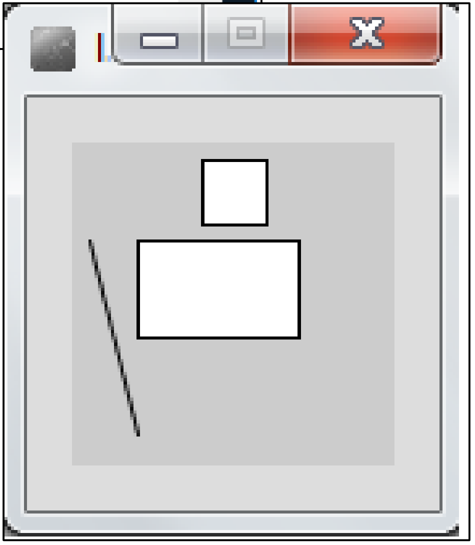
line () – drawing a line

Click to Run

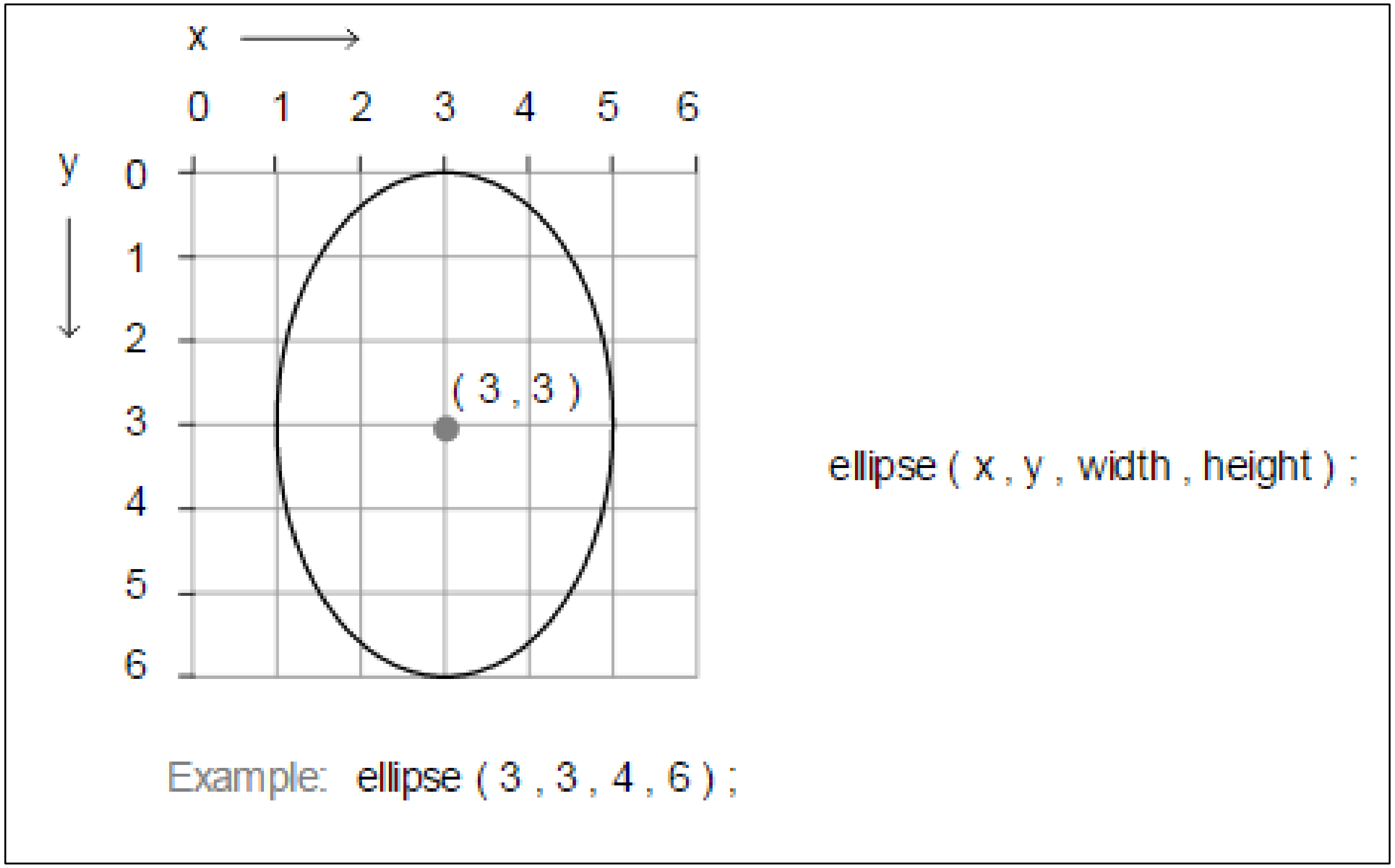

sketch_180103a | Processing 3.3.6
File Edit Sketch Debug Tools Help
Java
sketch_180103a

```
rect(20,30,50,30);
rect(40,5,20,20);
line(5,30,20,90);
```





ellipse()

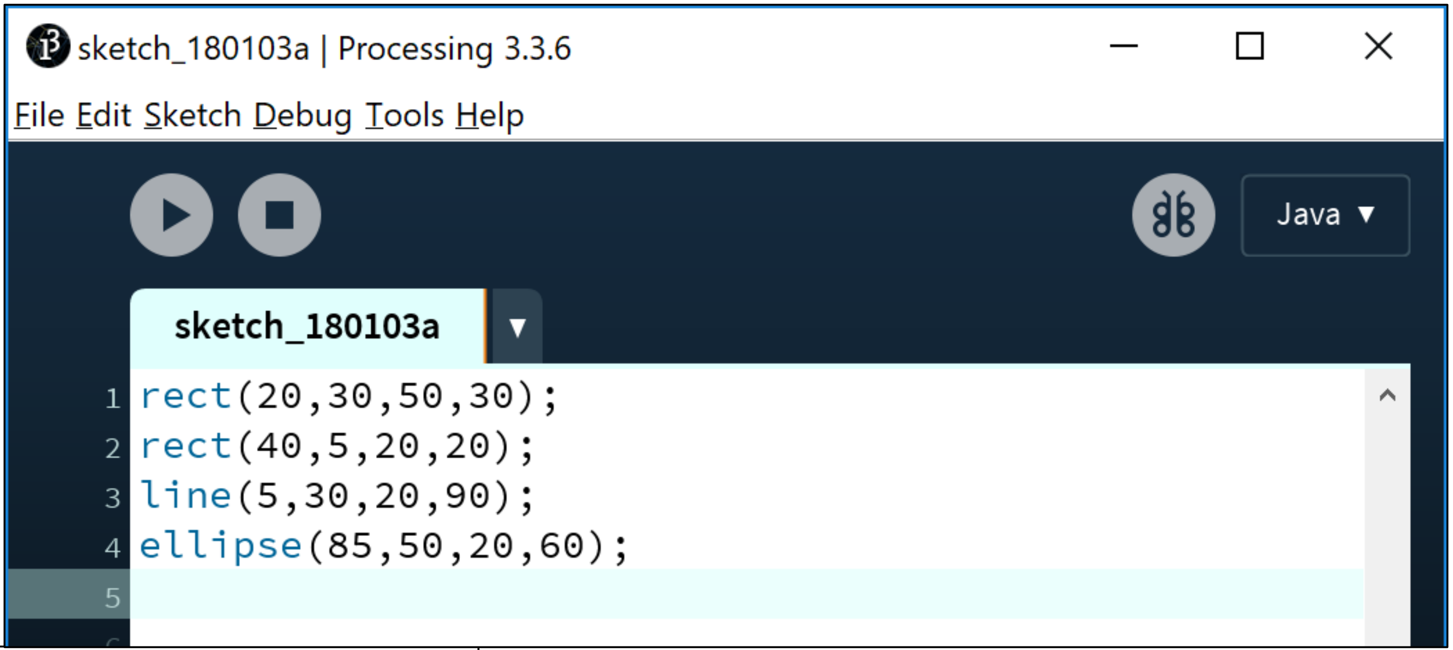


ellipse (x , y , width , height) ;

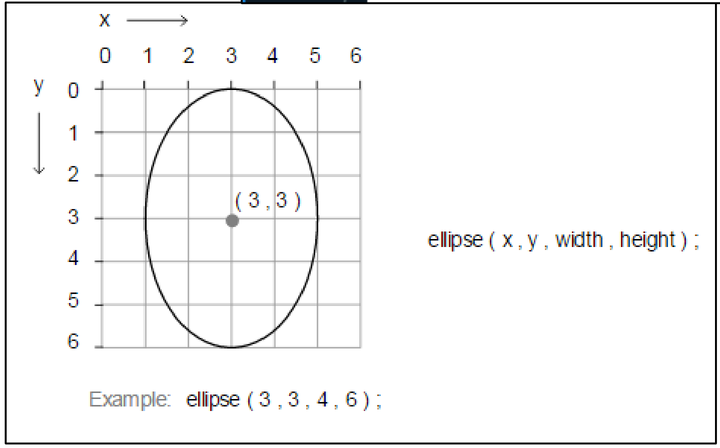
Example: ellipse (3 , 3 , 4 , 6) ;

https://processing.org/

ellipse() – drawing an oval

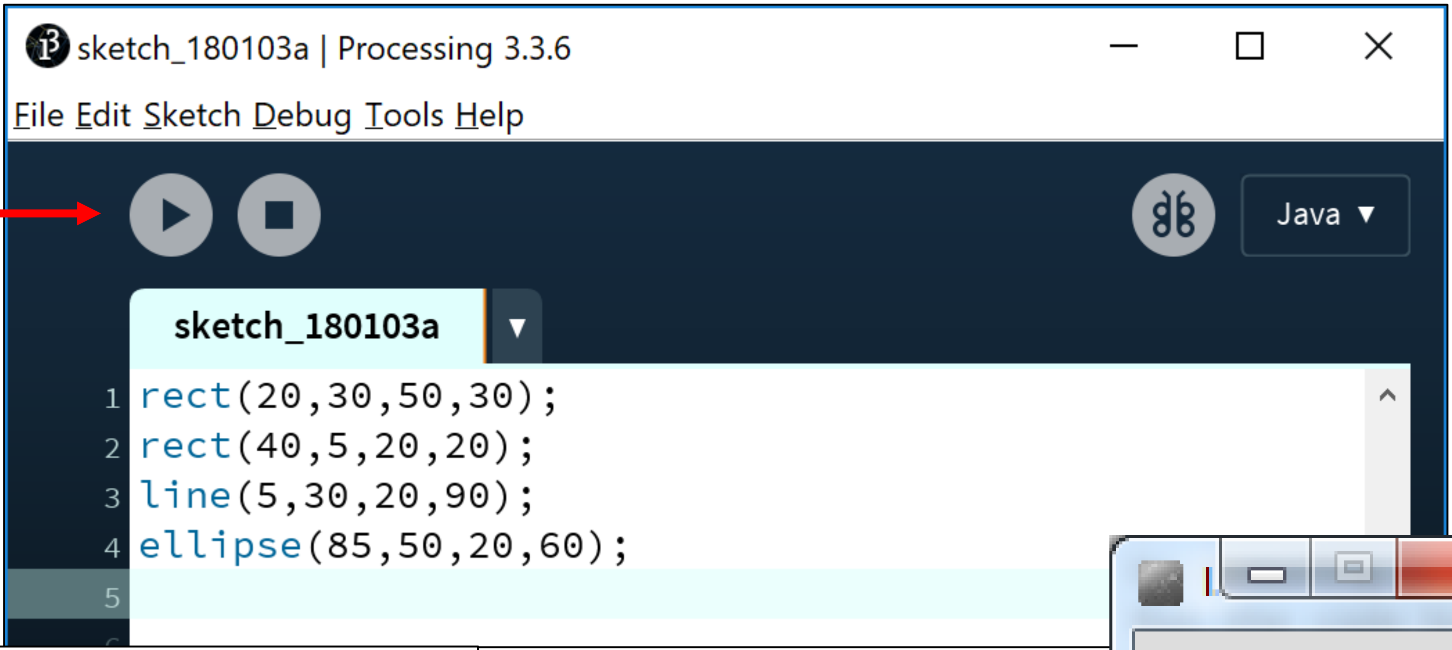

sketch_180103a | Processing 3.3.6
File Edit Sketch Debug Tools Help
Java
sketch_180103a

```
rect(20,30,50,30);
rect(40,5,20,20);
line(5,30,20,90);
ellipse(85,50,20,60);
```


x
0 1 2 3 4 5 6
y
0 1 2 3 4 5 6
(3 , 3)
ellipse (x , y , width , height) ;
Example: ellipse (3 , 3 , 4 , 6) ;

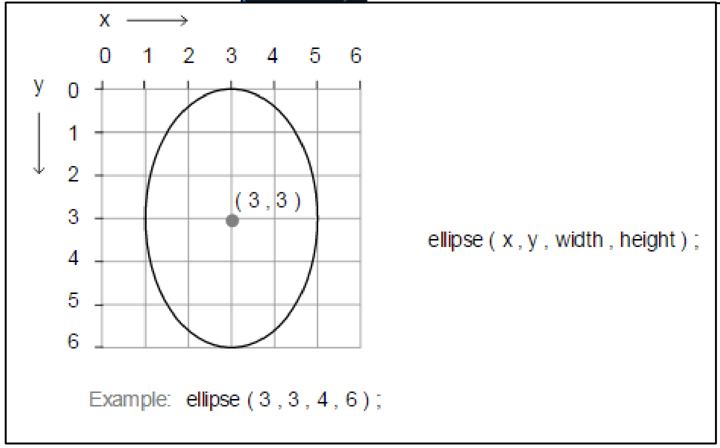
ellipse() – drawing an oval

Click to Run



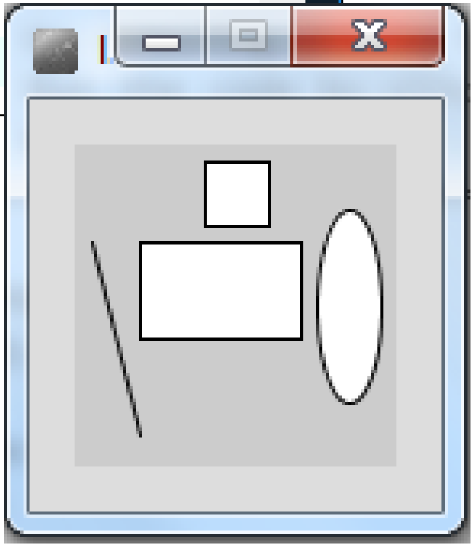
sketch_180103a | Processing 3.3.6
File Edit Sketch Debug Tools Help
Java
sketch_180103a

```
rect(20,30,50,30);
rect(40,5,20,20);
line(5,30,20,90);
ellipse(85,50,20,60);
```

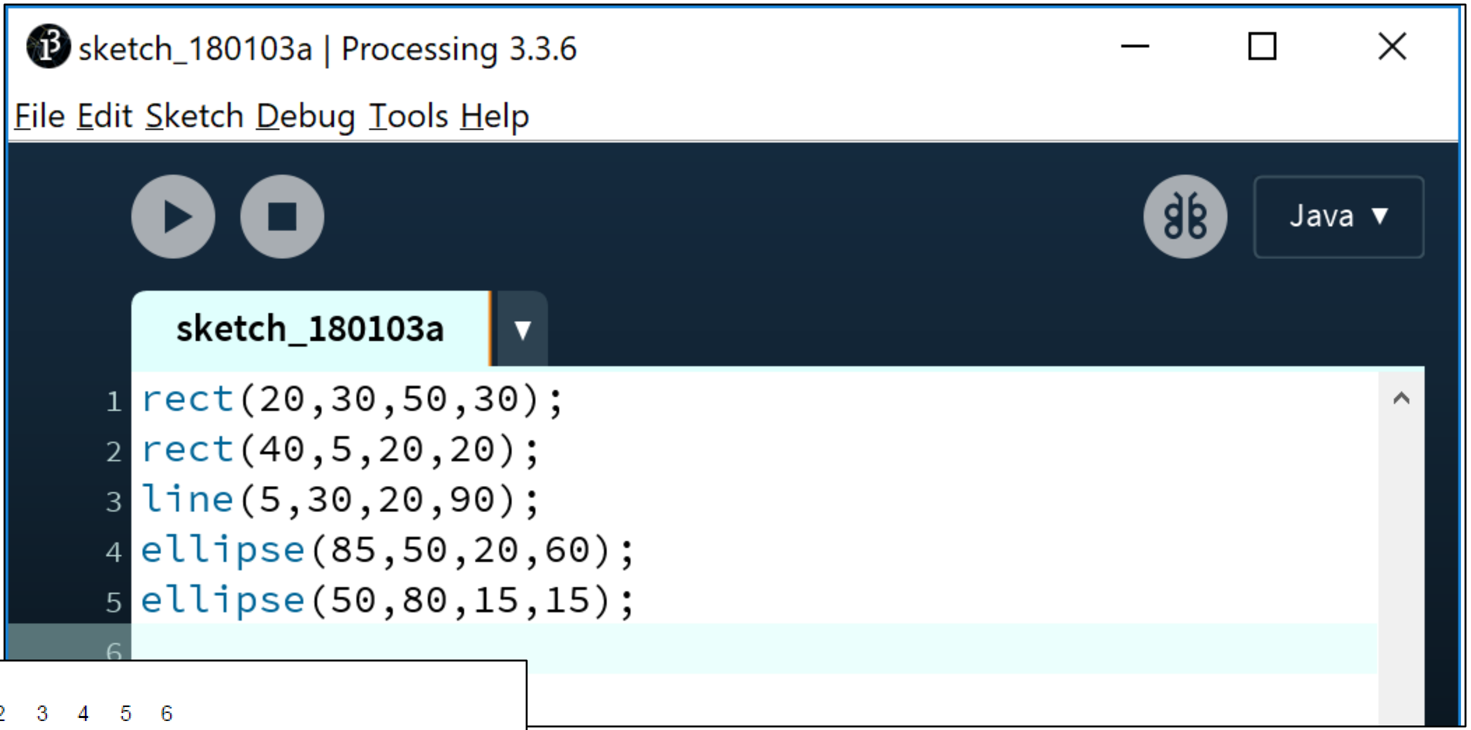


ellipse (x , y , width , height) ;

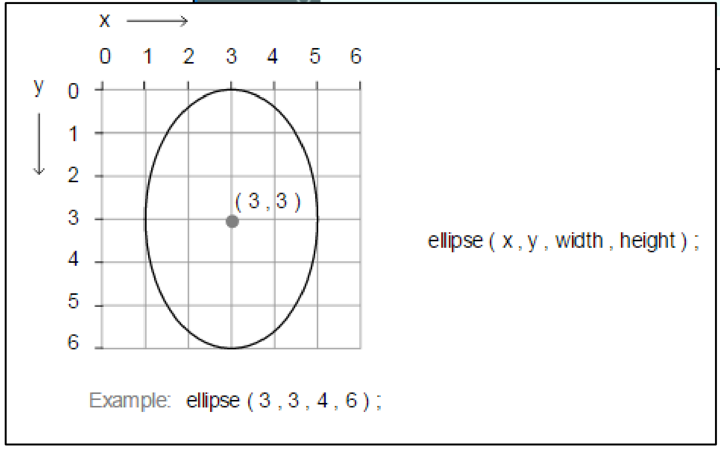
Example: ellipse (3 , 3 , 4 , 6) ;



ellipse() – drawing a circle


sketch_180103a | Processing 3.3.6
File Edit Sketch Debug Tools Help
Java
sketch_180103a

```
rect(20,30,50,30);
rect(40,5,20,20);
line(5,30,20,90);
ellipse(85,50,20,60);
ellipse(50,80,15,15);
```

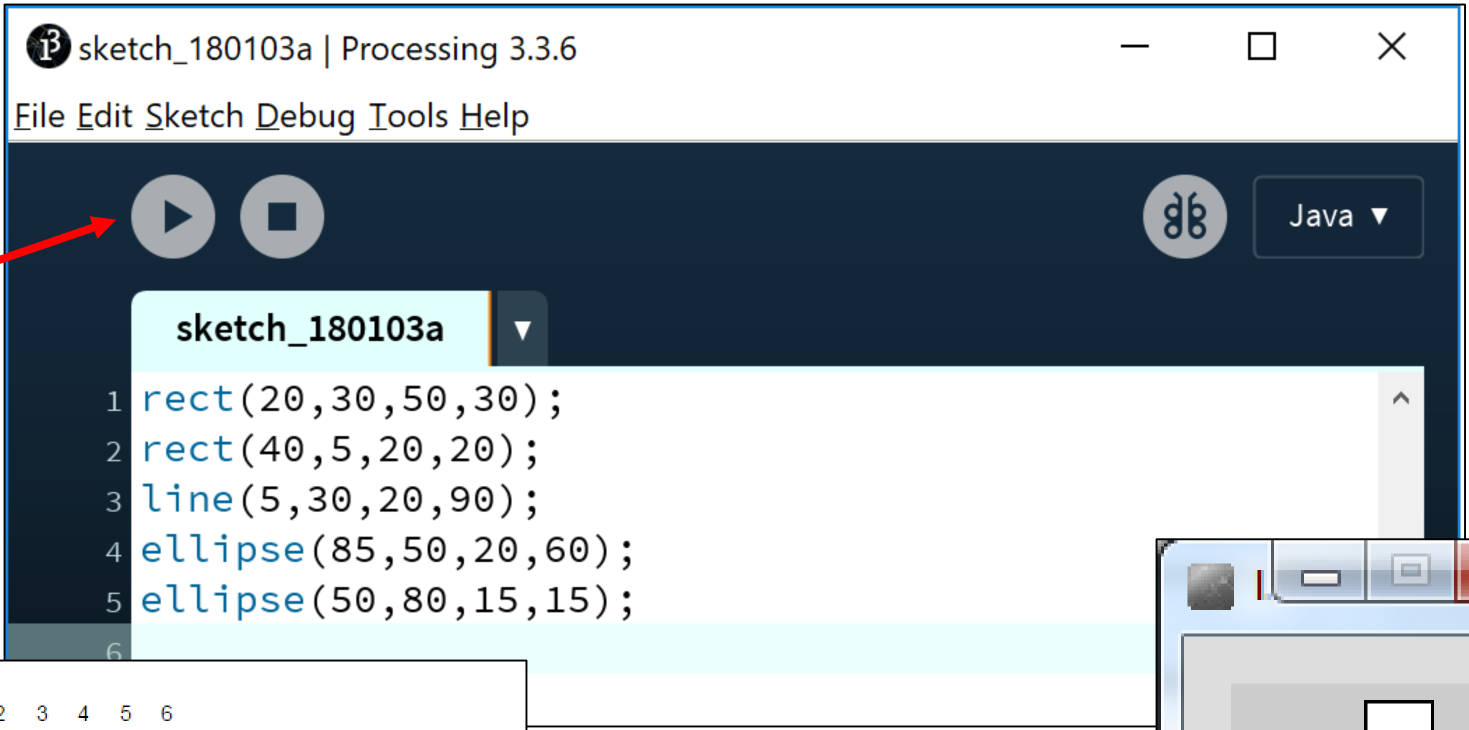


ellipse (x , y , width , height) ;

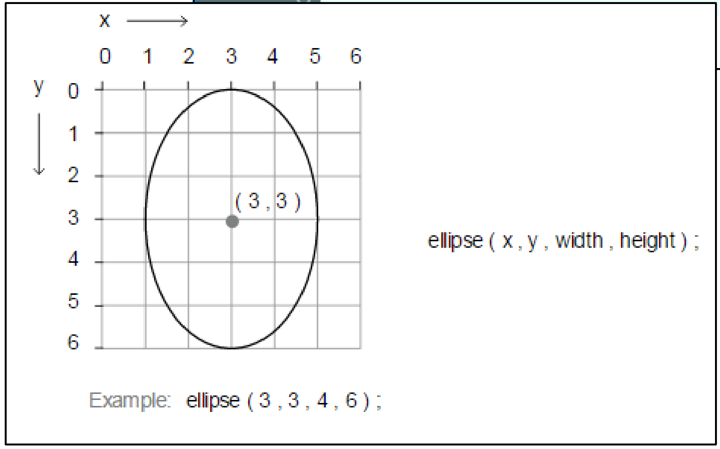
Example: ellipse (3 , 3 , 4 , 6) ;

ellipse() – drawing a circle

Click to Run

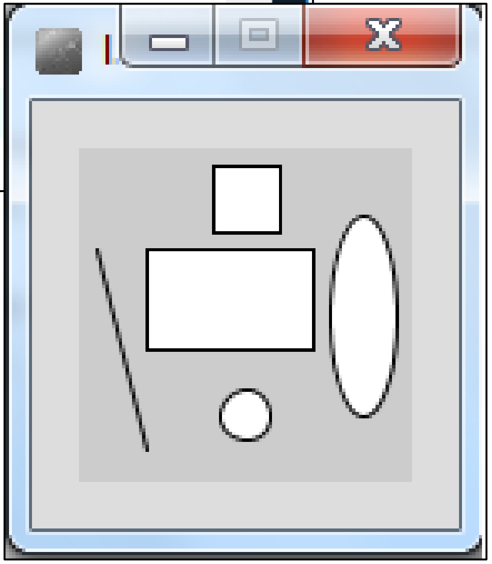

sketch_180103a | Processing 3.3.6
File Edit Sketch Debug Tools Help
Java
sketch_180103a

```
rect(20,30,50,30);
rect(40,5,20,20);
line(5,30,20,90);
ellipse(85,50,20,60);
ellipse(50,80,15,15);
```



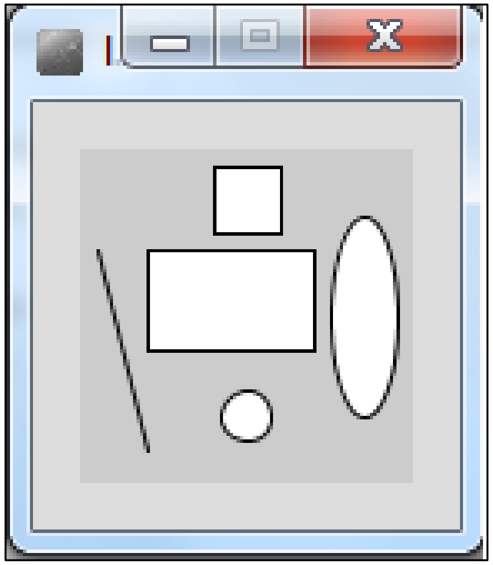
ellipse (x , y , width , height) ;

Example: ellipse (3 , 3 , 4 , 6) ;



Formatting the display window

- Our display window is looking fairly cramped.
- The default size of your display window is 100x100 pixels, which is quite small.

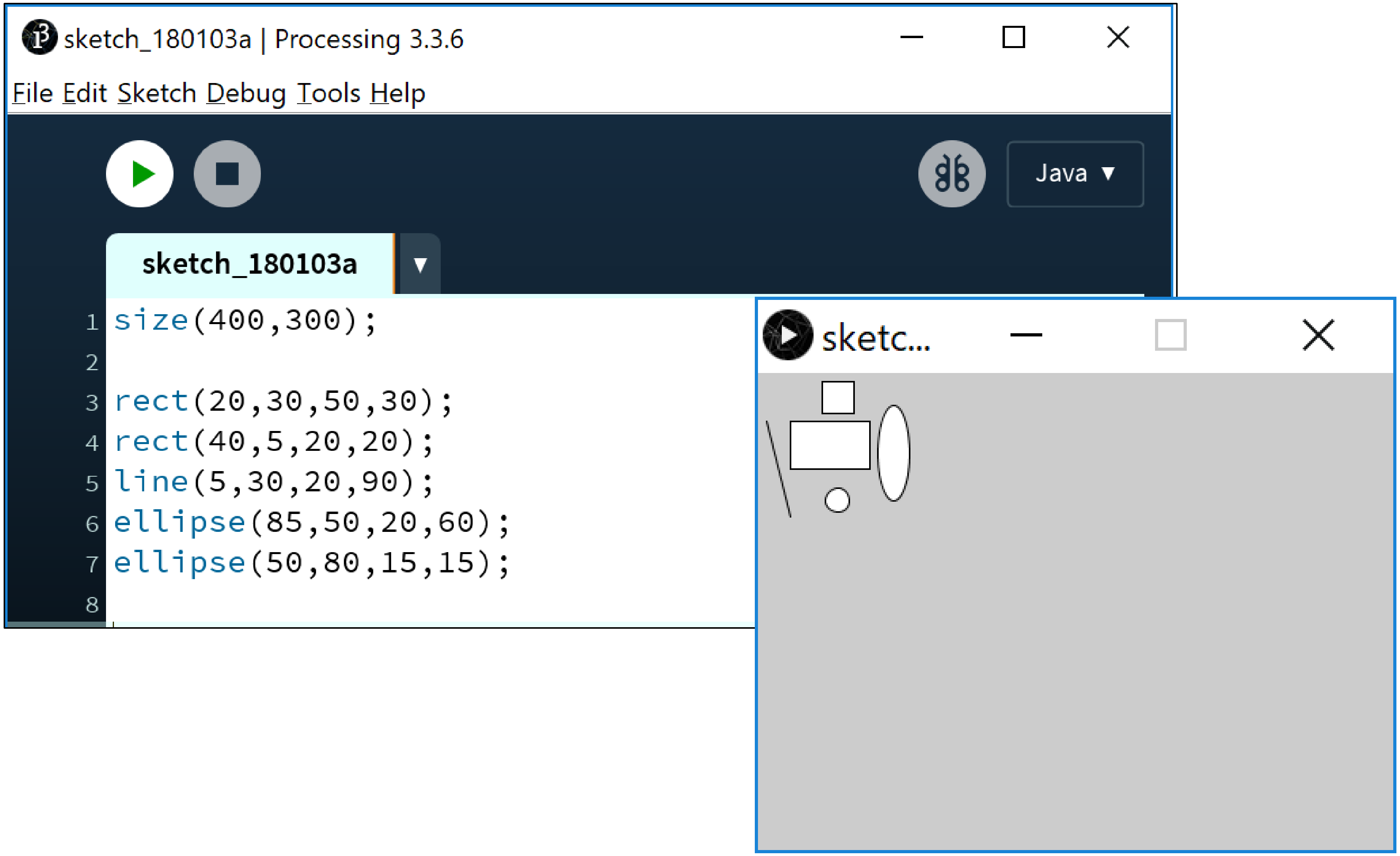


Formatting the display window

- We can change the size of the display window by calling the size function.
- When you use the size function in static drawings, it has to be the first line of code in your sketchbook.

```
size(w, h)
    w = width of the display window
    h = height of the display window
```

size()

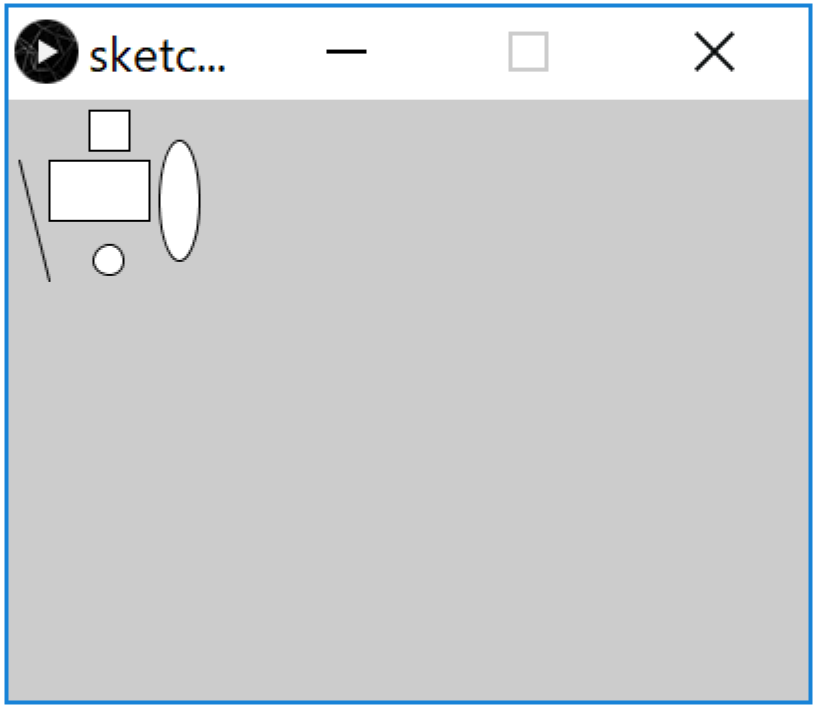

sketch_180103a | Processing 3.3.6

File Edit Sketch Debug Tools Help

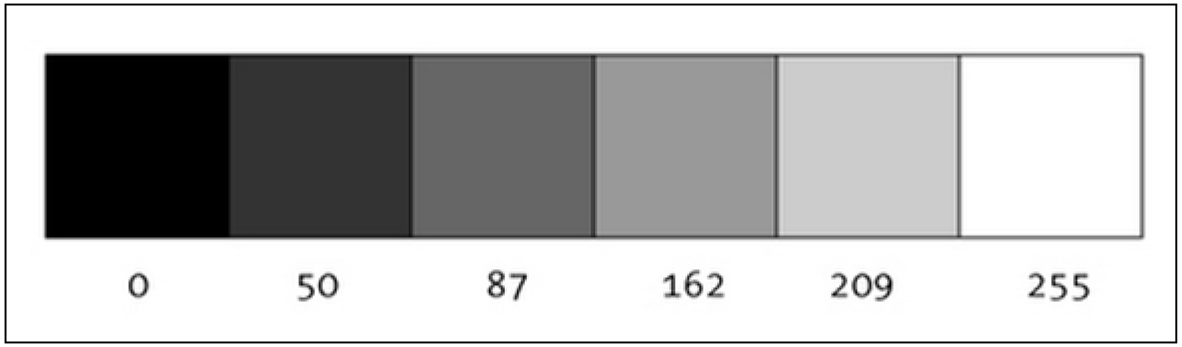
```
size(400,300);

rect(20,30,50,30);
rect(40,5,20,20);
line(5,30,20,90);
ellipse(85,50,20,60);
ellipse(50,80,15,15);
```

Formatting the display window

- Our display window looks less cramped now.
- But maybe we want to change the default gray colour?
- We could use the background function to set the colour to something else.



A note on colour first…Grayscale



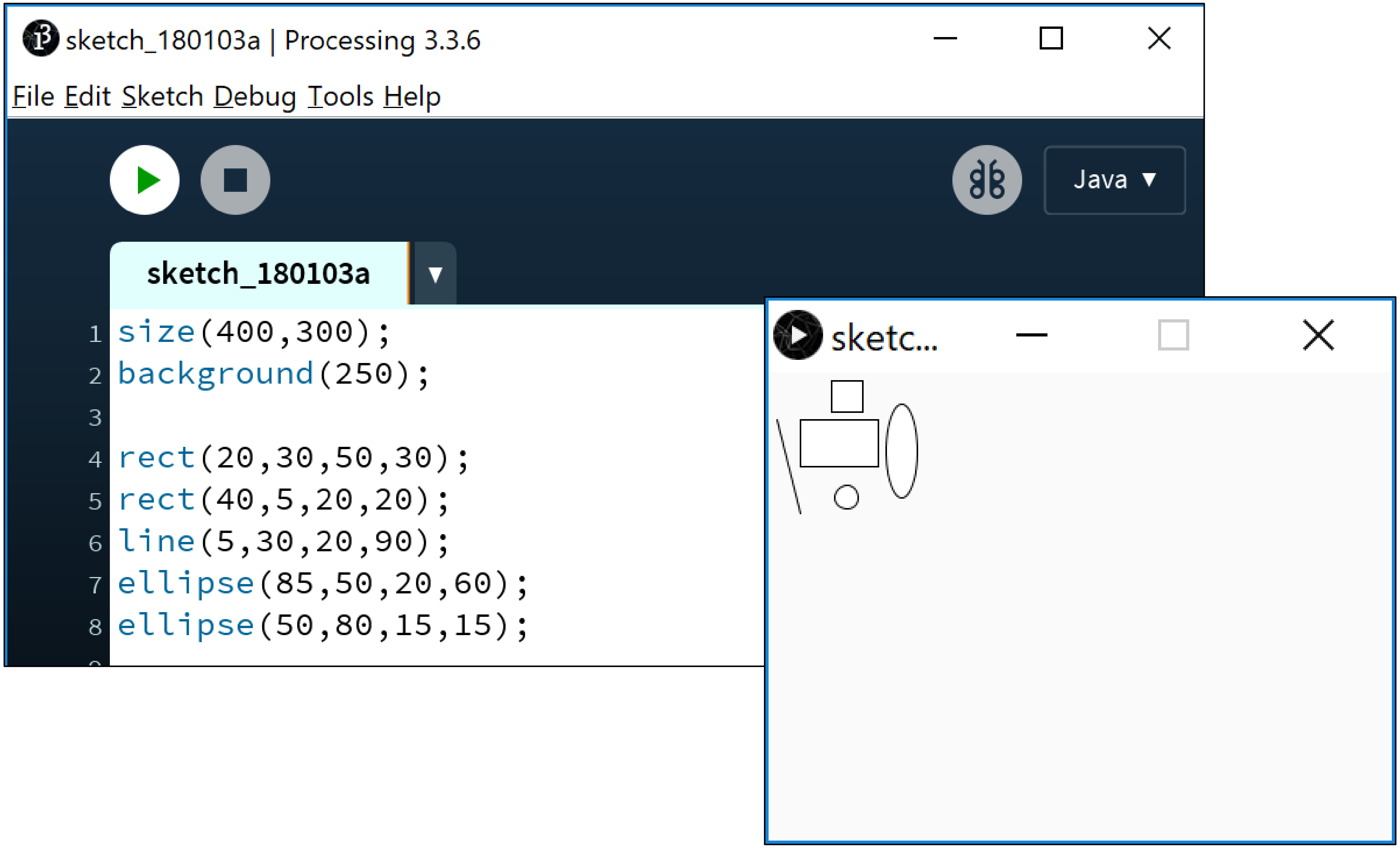
“0 means black, 255 means white. In between, every other number - 50, 87, 162, 209, and so on - is a shade of gray ranging from black to white.”

https://www.processing.org/tutorials/color/

background() - syntax

```
background(grayscale)

    grayscale = grayscale colour (a number between
                0 [black] and 255 [white] inclusive)
```

background()


sketch_180103a | Processing 3.3.6
File Edit Sketch Debug Tools Help
Java
sketch_180103a
sketc...

```
size(400,300);
background(250);

rect(20,30,50,30);
rect(40,5,20,20);
line(5,30,20,90);
ellipse(85,50,20,60);
ellipse(50,80,15,15);
```

Questions?