

# An Introduction to Processing

---

Creating Static Drawings

Lecturer: Caio Fonseca



Waterford Institute *of* Technology  
INSTITIÚID TEICNEOLAÍOCHTA PHORT LÁIRGE

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

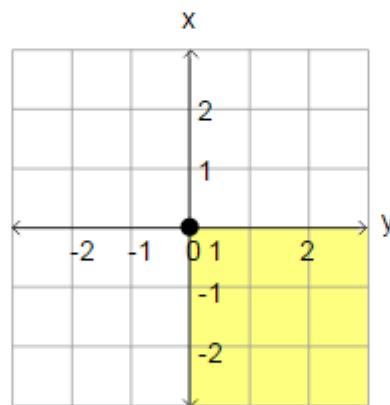
# Topics list

---

- Coordinate System in Computing.
- Functions in Processing.
- Basic Shapes.
- Formatting the display window.
- Syntax Errors.
- Logic Errors.

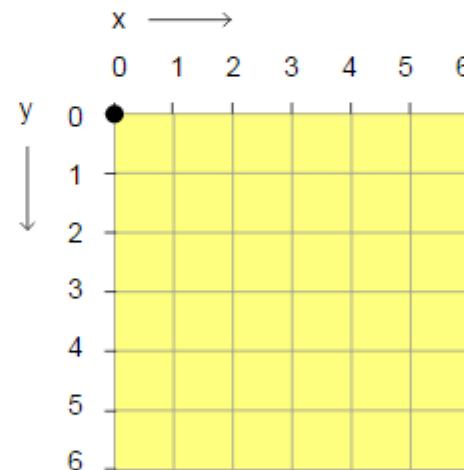
# Coordinate System in Computing

In Geometry,  
we use this type of  
coordinate system:



point (0,0) is in the centre.

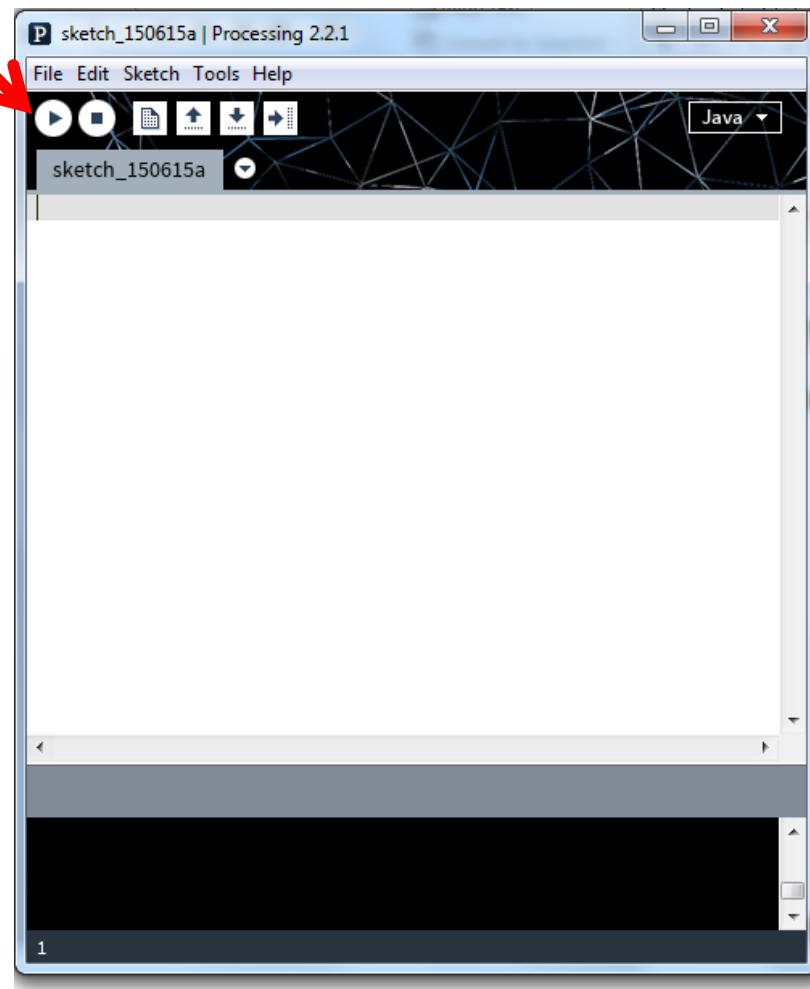
In Computing, we use this type of coordinate system to represent the screen:



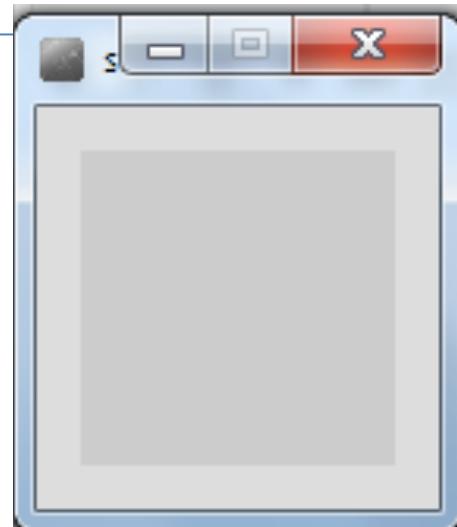
point (0,0) is in the top left hand corner. Each number is a pixel.

# Coordinate System in Computing

Run  
button



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

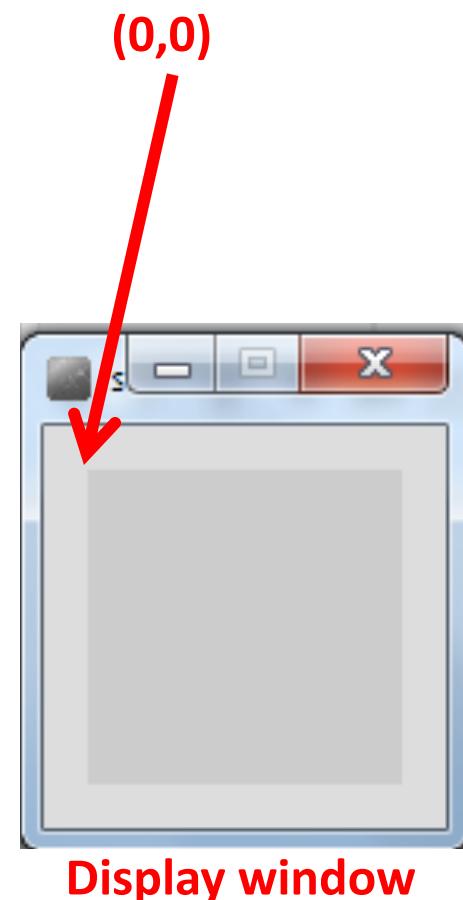


Display window

# 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).



# Topics list

---

- Coordinate System in Computing.
- Functions in Processing.
- Basic Shapes.
- Formatting the display window.
- Syntax Errors.
- Logic Errors.

# Functions in Processing

---

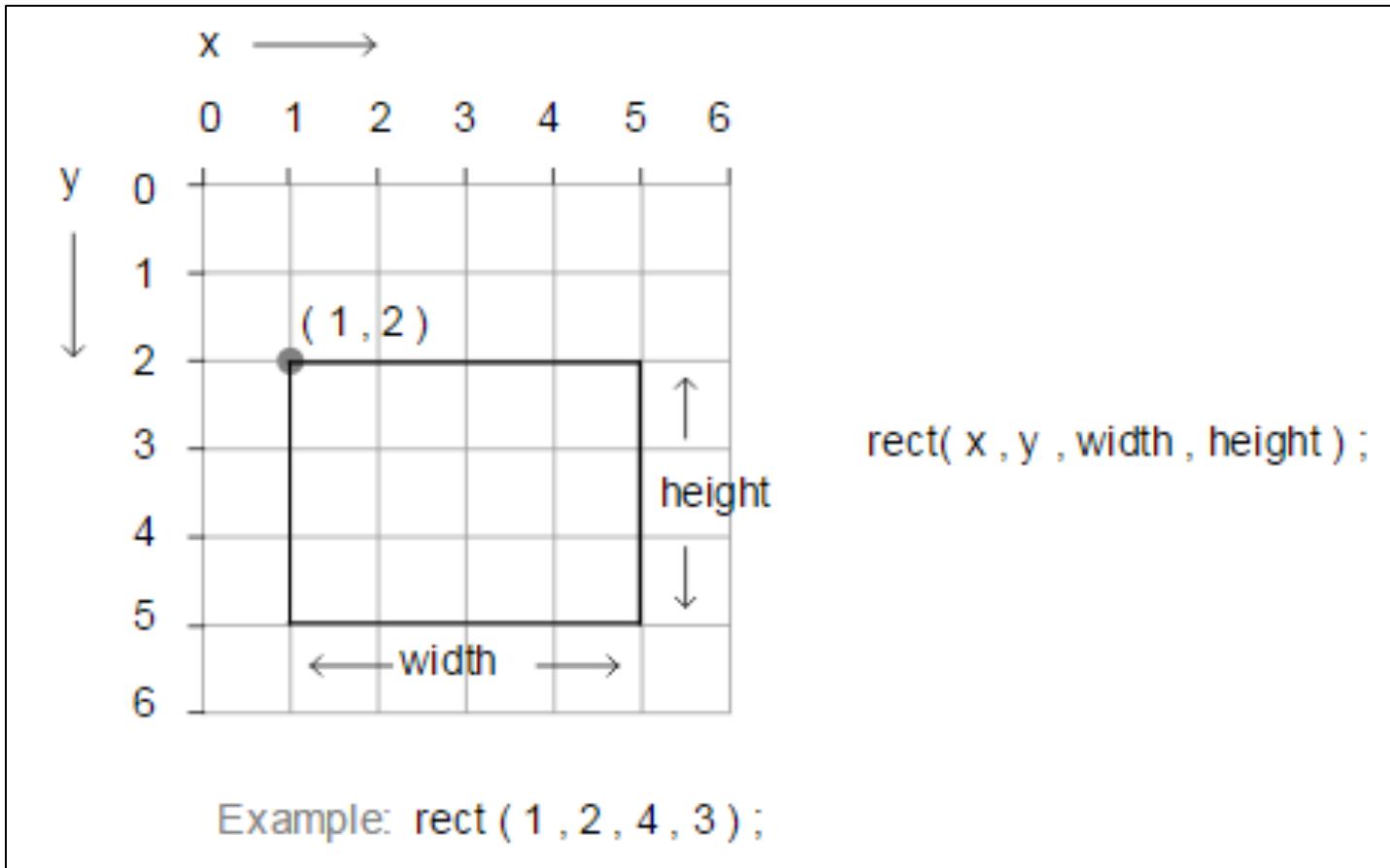
- Processing comes with several pre-written functions that we can use.
- A function comprises a set of instructions that performs some task.
- When you call the function, it performs the task.
- We will now look at functions that draw the following shapes:
  - Rectangle, square, line, oval and circle.

# Topics List

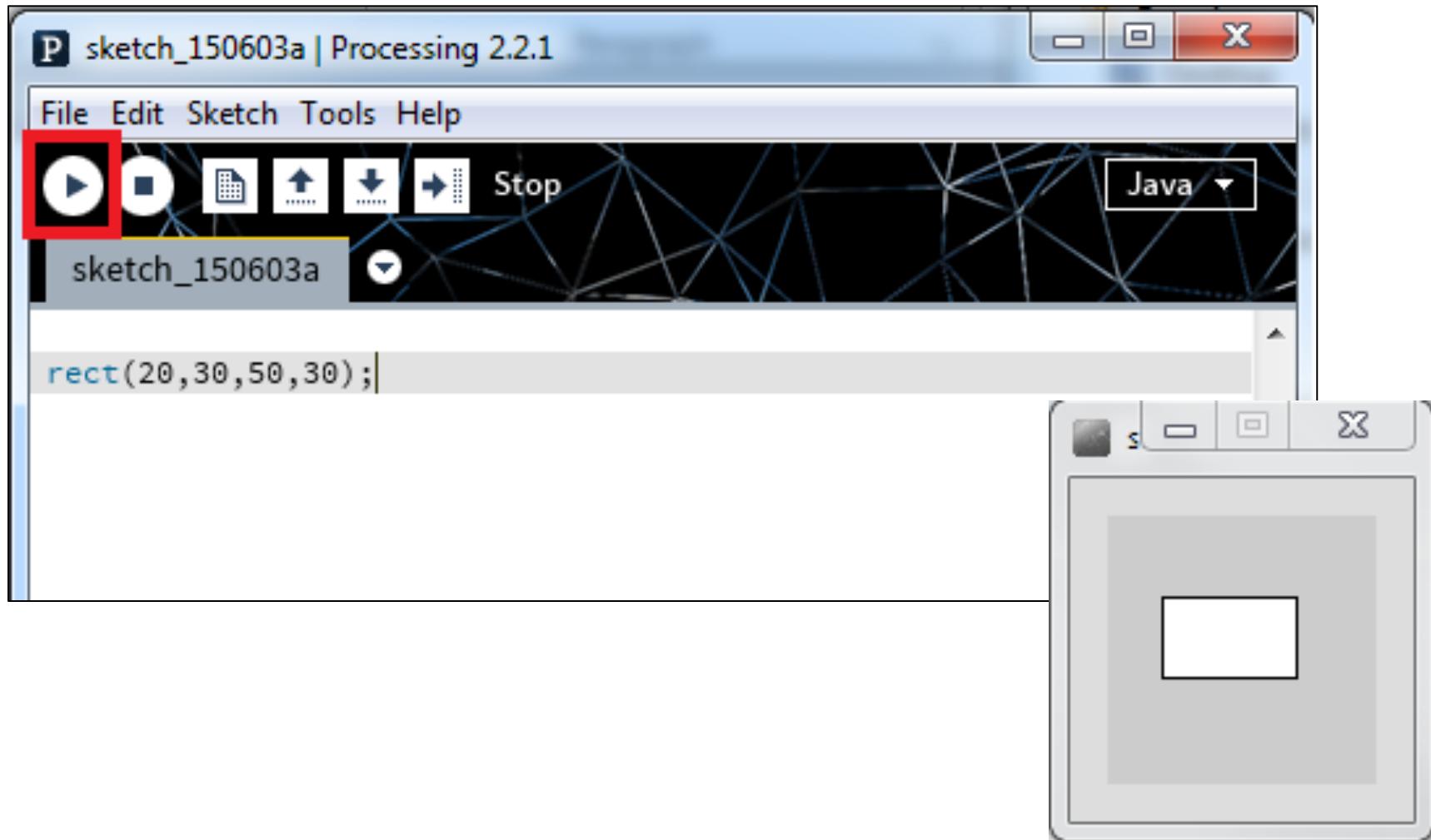
---

- Coordinate System in Computing.
- Functions in Processing.
- Basic Shapes.
- Formatting the display window.
- Syntax Errors.
- Logic Errors.

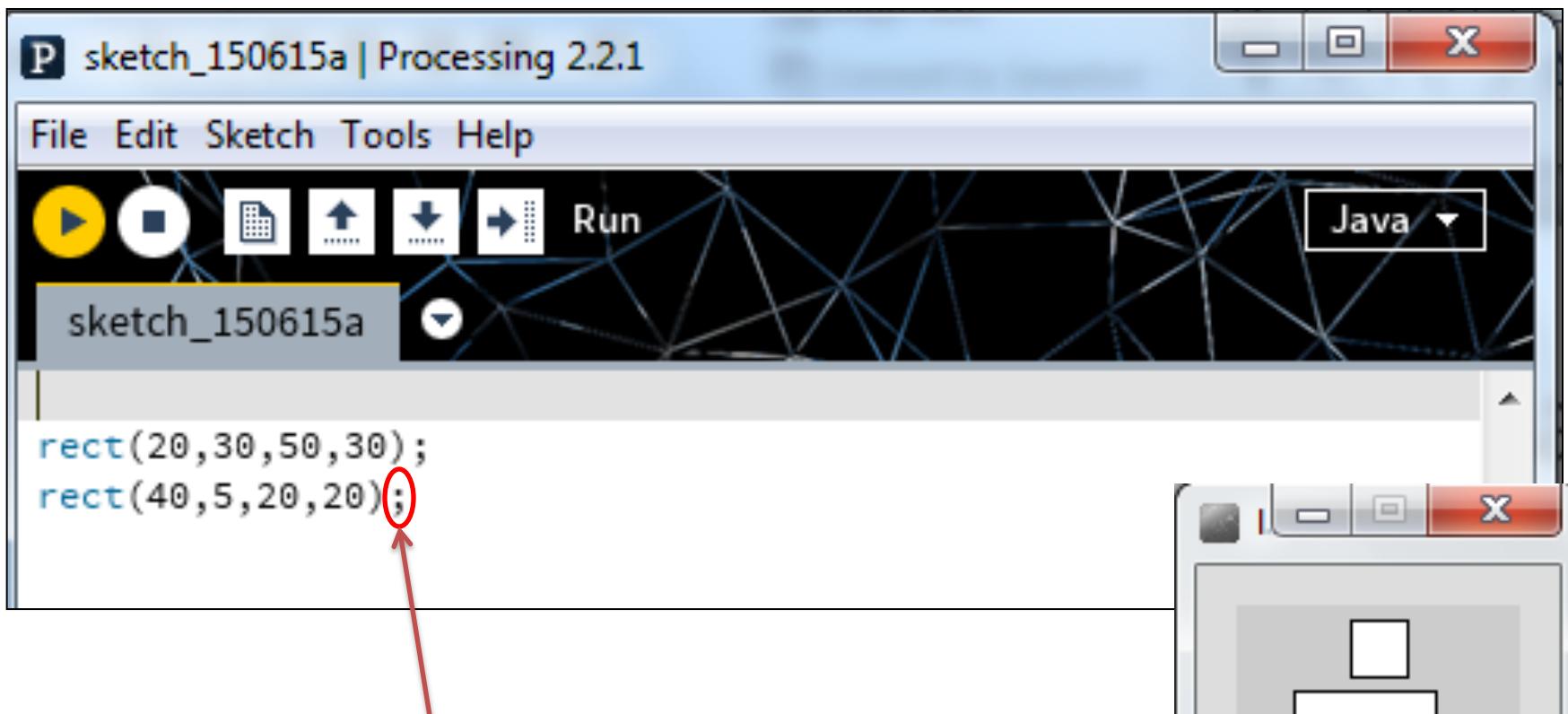
# rect()



# rect() – Drawing a Rectangle



# rect() – Drawing a Square

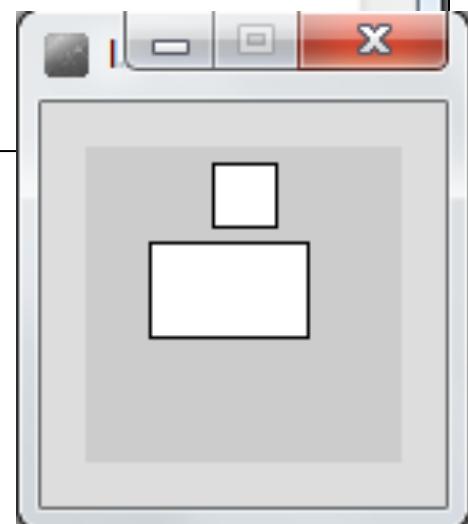


The screenshot shows the Processing 2.2.1 IDE interface. The title bar says "P sketch\_150615a | Processing 2.2.1". The menu bar includes File, Edit, Sketch, Tools, and Help. The toolbar has icons for play, stop, file, upload, download, and run. A dropdown menu shows "Java". The code editor contains the following code:

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

A red circle highlights the semi-colon at the end of the second line of code, and a red arrow points from the text below to this circled character.

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.



# rect() – Syntax

---

`rect(x, y, w, h)`

`x` = x-coordinate of the upper left corner of the rectangle

`y` = y-coordinate of the upper left corner of the rectangle

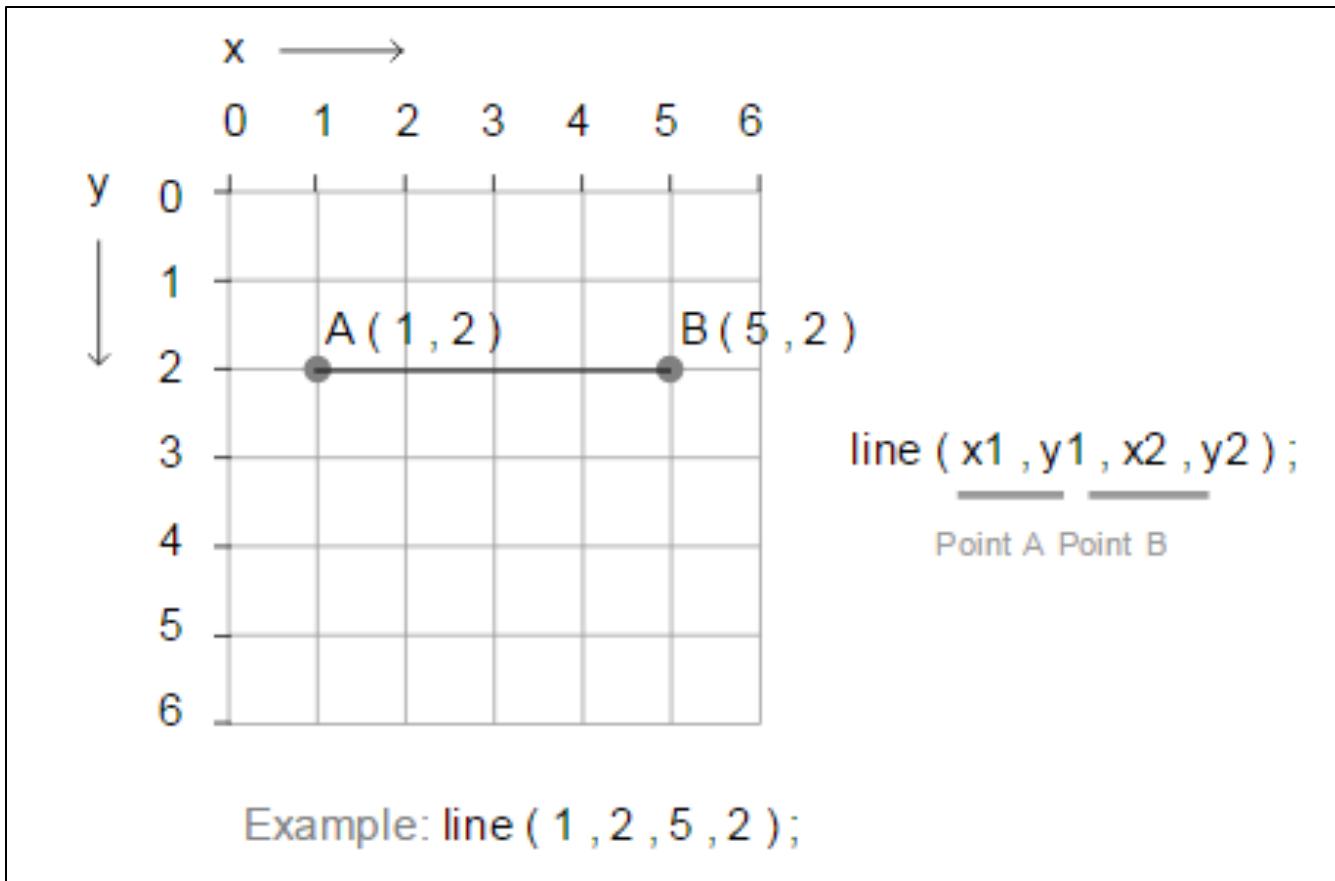
`w` = width of the rectangle

`h` = height of the rectangle

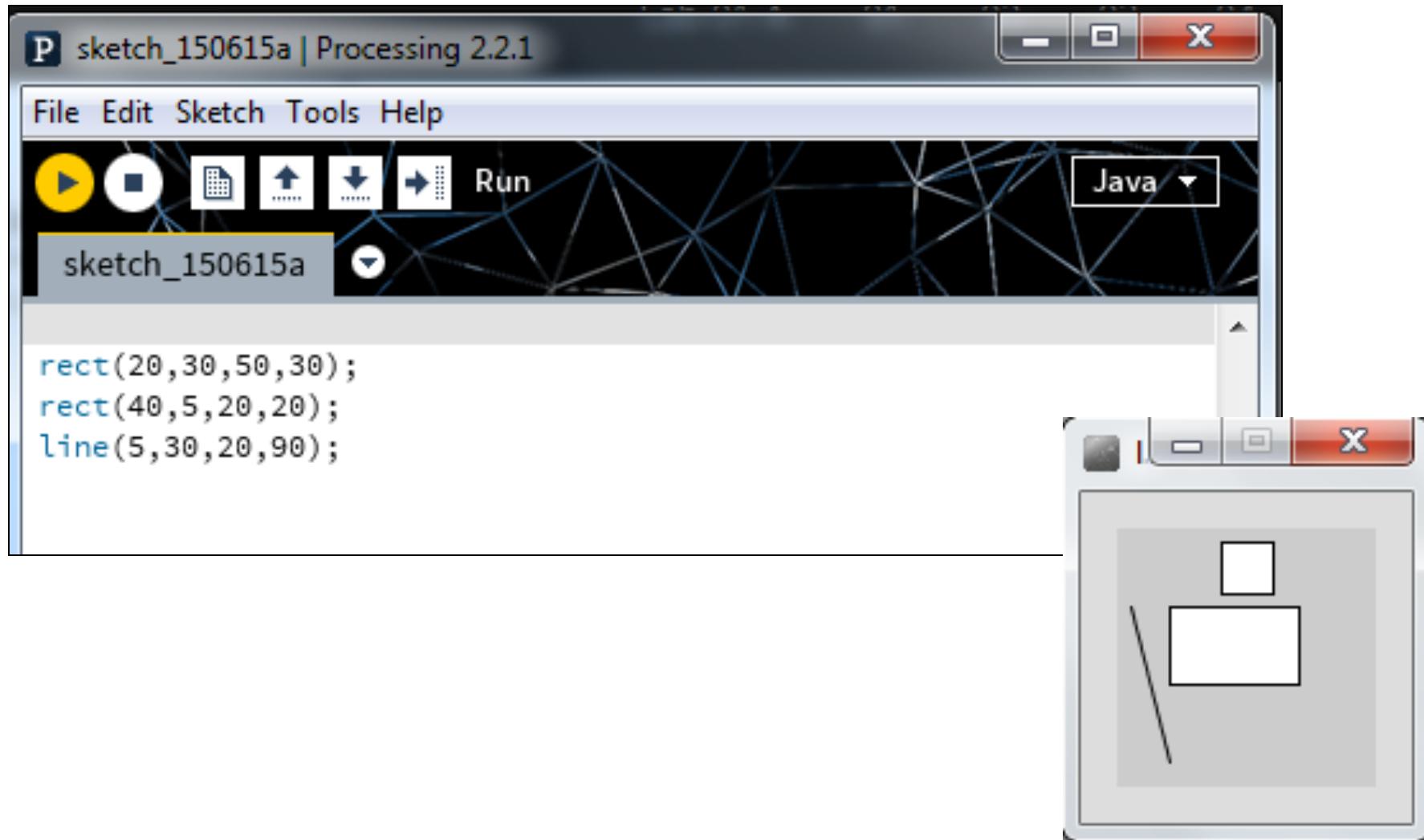
- The rect function above defines four **parameters** i.e. `x, y, w, h`.
- When you call rect, you are expected to pass four numbers to it. These actual numbers are called **arguments**.
- rect uses these four numbers to render the rectangle on the display window.

To draw a square, the width and height must be the same value.

# line()



# line() – Drawing a Line



# line() – Syntax

---

**line(x1, y1, x2, y2)**

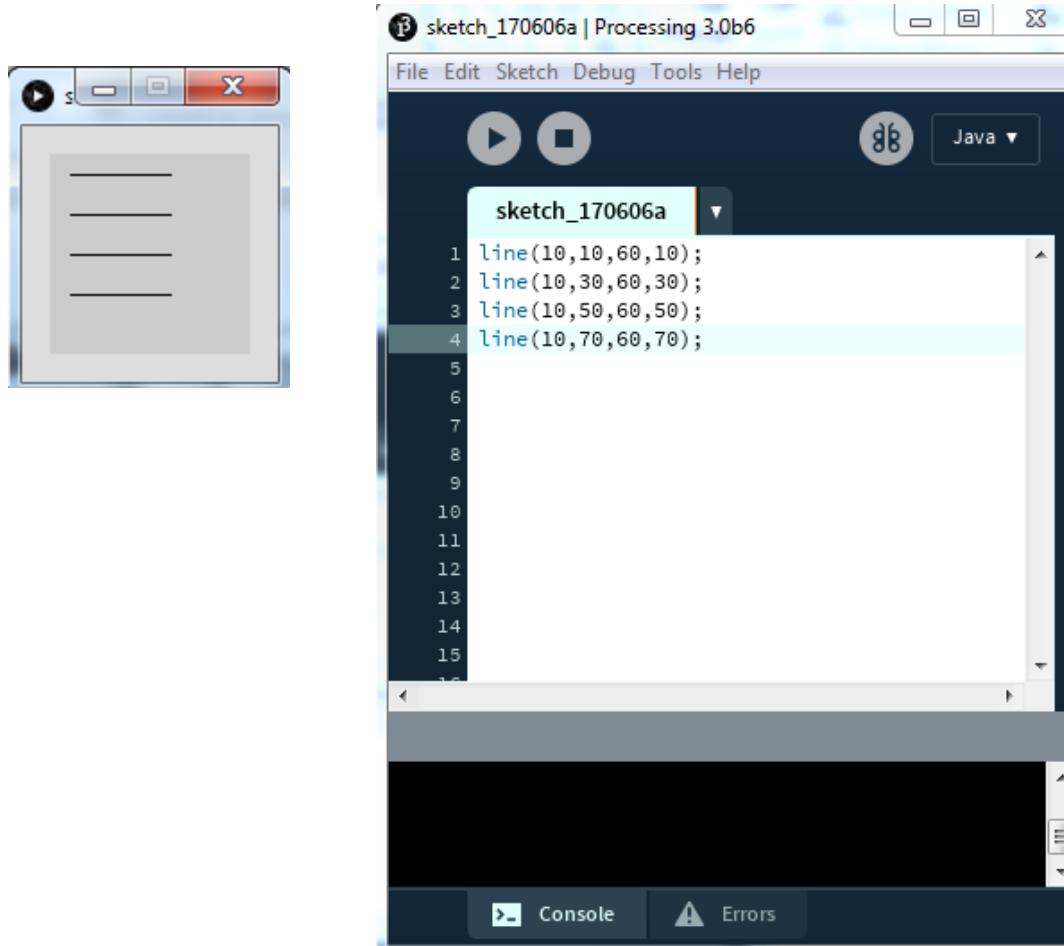
x1 = x-coordinate of first point

y1 = y-coordinate of first point

x2 = x-coordinate of second point

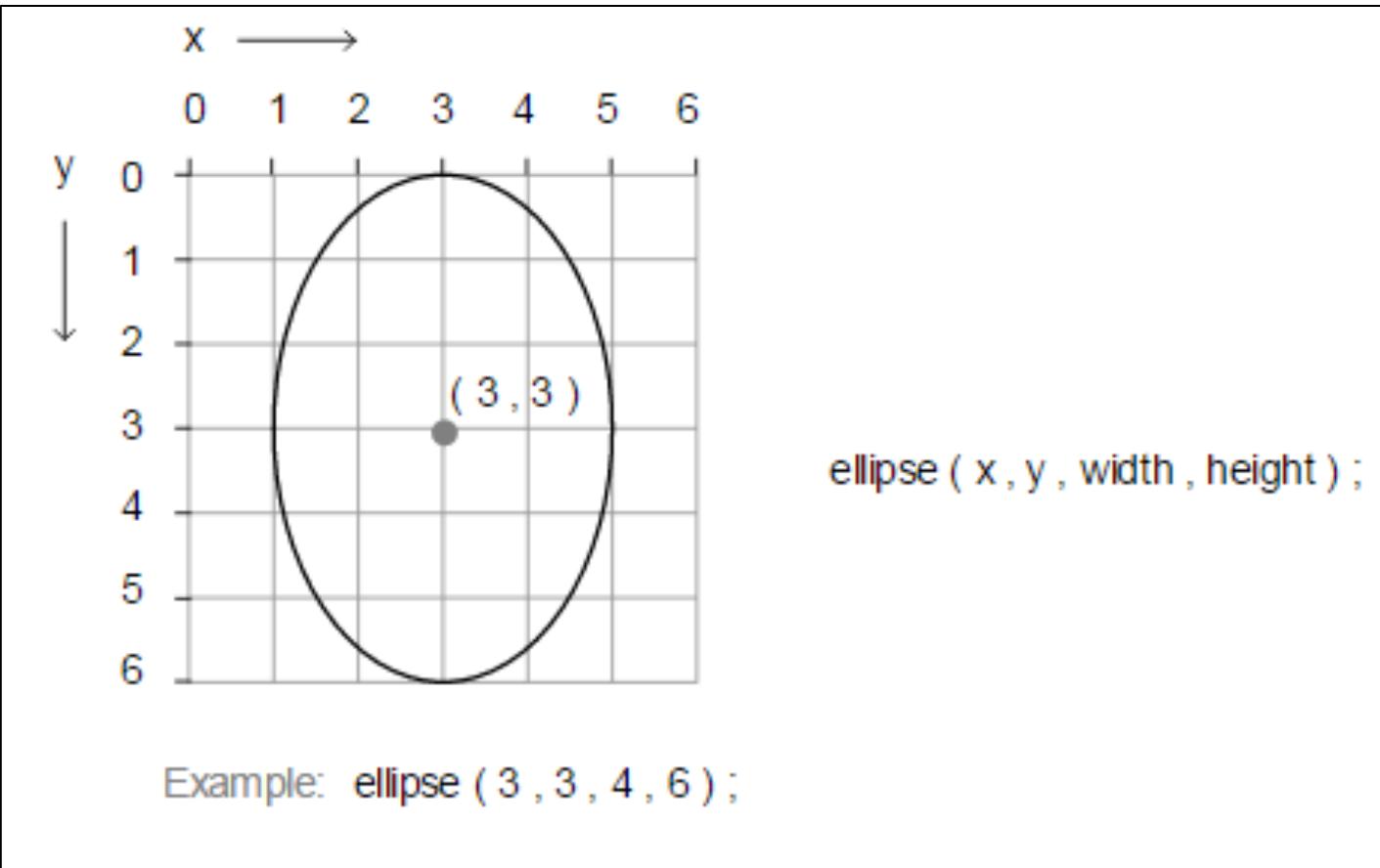
y2 = y-coordinate of second point

# line() – More Examples

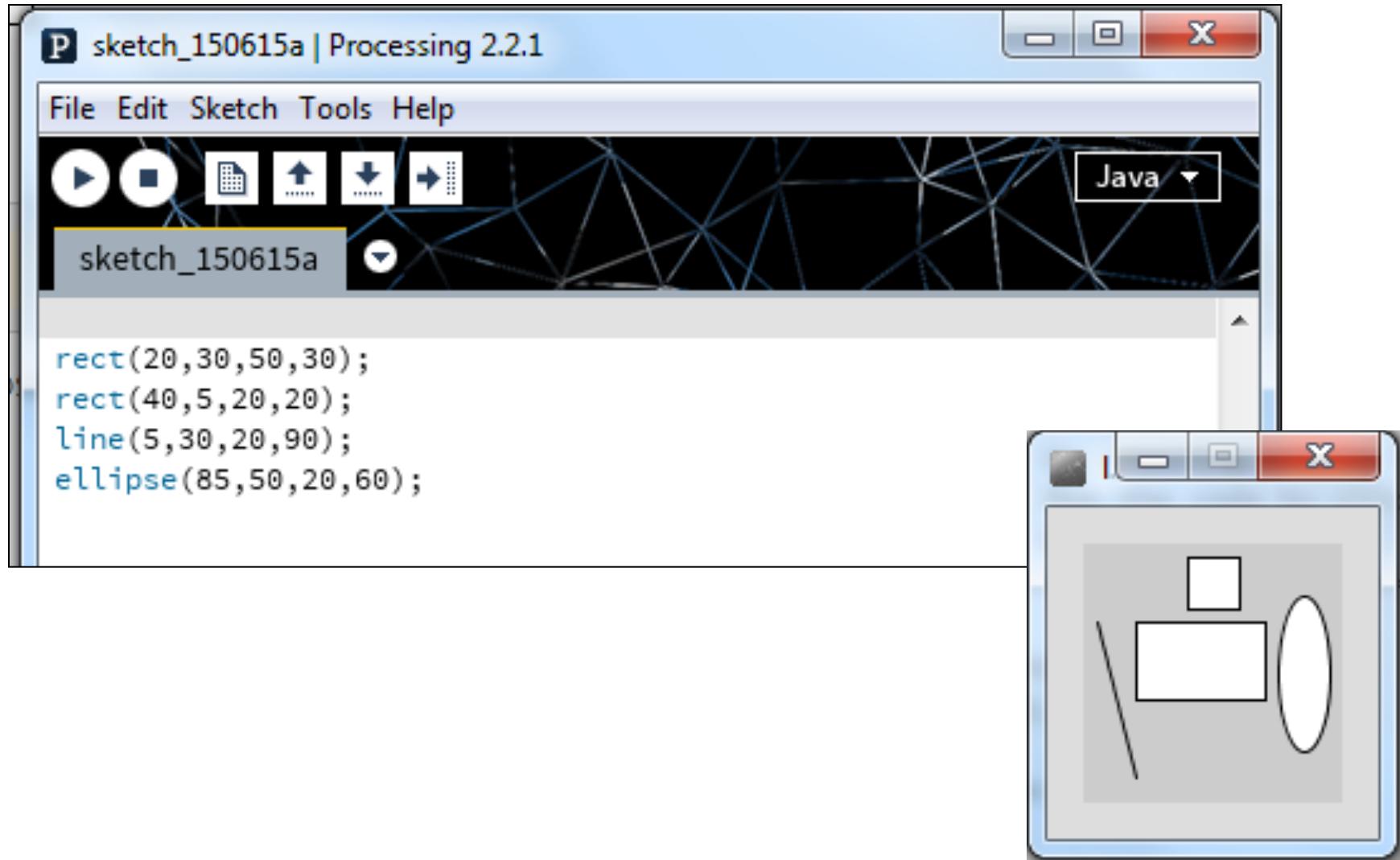


# ellipse()

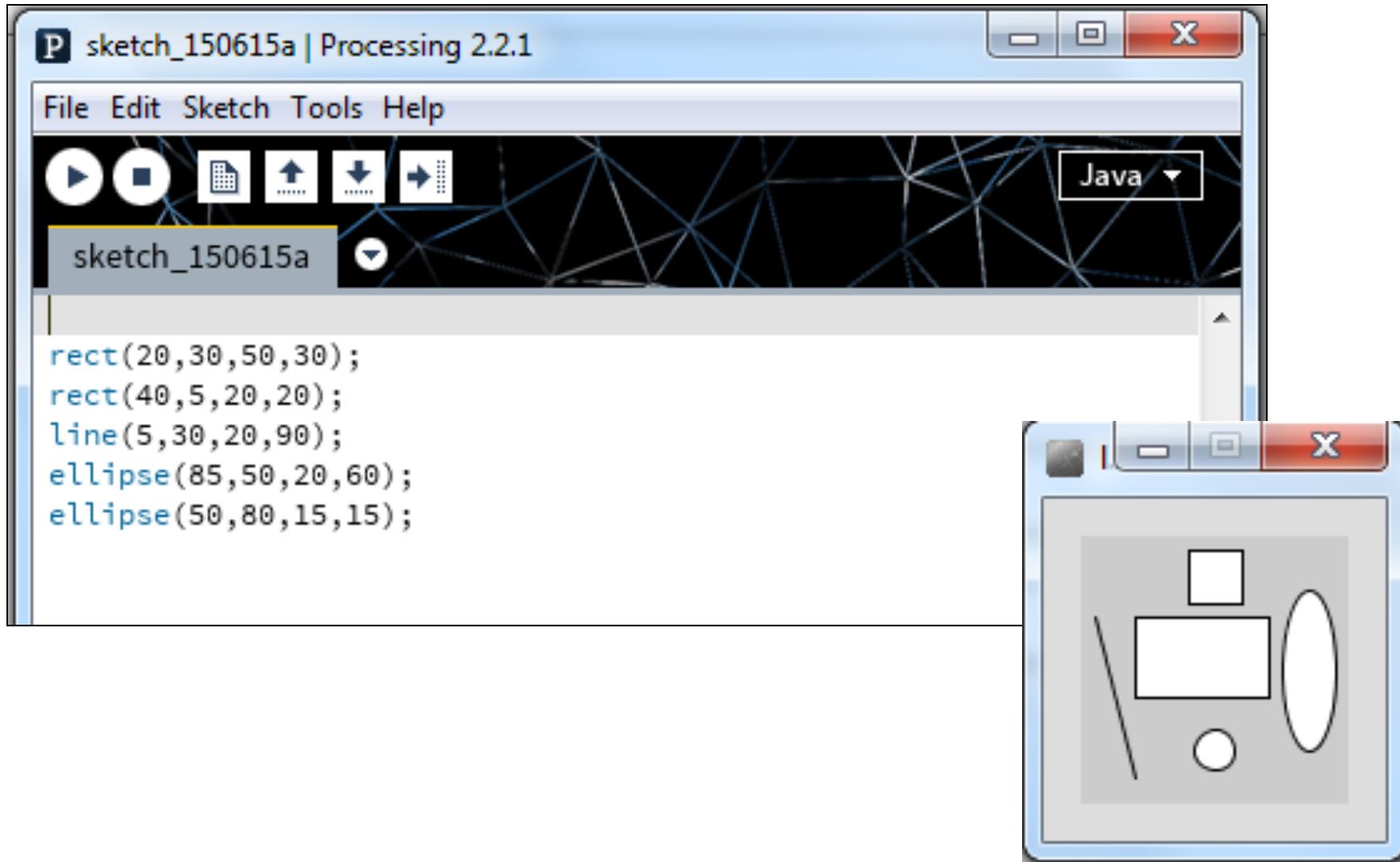
---



# ellipse()



# ellipse()



The image shows the Processing 2.2.1 software interface. The top bar displays "P sketch\_150615a | Processing 2.2.1". The menu bar includes File, Edit, Sketch, Tools, and Help. Below the menu is a toolbar with icons for play, stop, file, and zoom. The central code editor window contains the following Java code:

```
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);
```

To the right of the code editor is a preview window titled "Java" which displays the resulting graphic output. The output shows a gray rectangular background with several black outlines: a large rectangle at [20, 30, 50, 30], a smaller rectangle at [40, 5, 20, 20], a line from [5, 30] to [20, 90], an ellipse at [85, 50, 20, 60] with a bounding box of approximately [85, 50, 105, 56], and another ellipse at [50, 80, 15, 15] with a bounding box of approximately [50, 80, 65, 95].

# ellipse() – Syntax

---

**ellipse(x, y, w, h)**

x = x-coordinate of the centre of the ellipse

y = y-coordinate of the centre of the ellipse

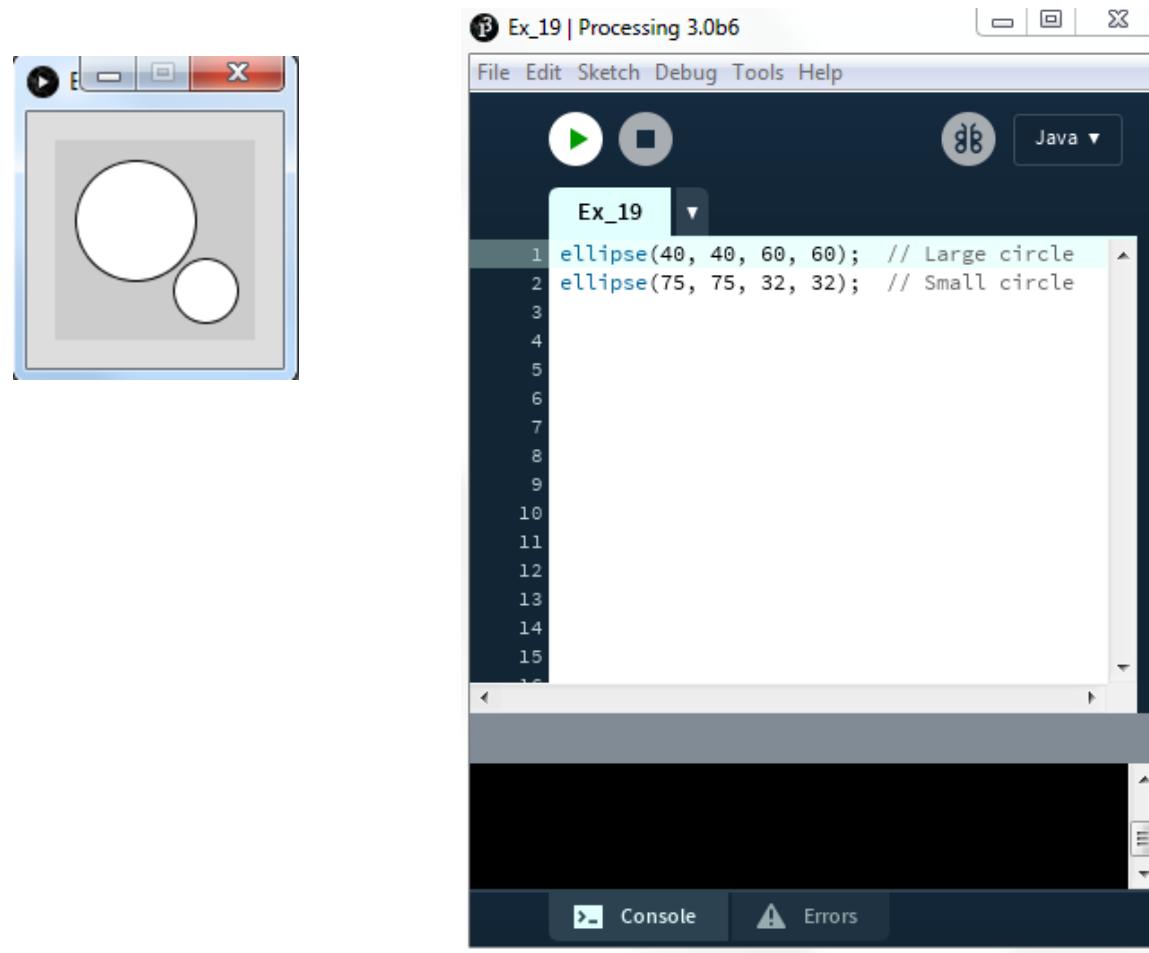
w = width of the ellipse

h = height of the ellipse

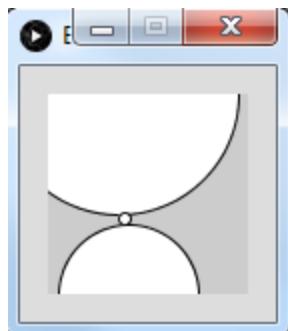
To draw a circle,  
the width and  
height must be the  
same value.

# ellipse() - More Examples

---



# ellipse() - More Examples



What's wrong with  
the ellipse here??

A screenshot of the Processing IDE showing the code editor and toolbars. The title bar says "Ex\_20 | Processing 3.0b6". The code editor shows the following Java code:

```
1 ellipse(35, 0, 120, 120); // first circle
2 ellipse(38, 62, 6, 6); // second circle
3 ellipse(40, 100, 70, 70); // third circle
4
5
6
7
8
9
10
11
12
13
14
15
16
```

The code editor has tabs for "Ex\_20" and "Java". Below the code editor are two tabs: "Console" and "Errors".

# Topics List

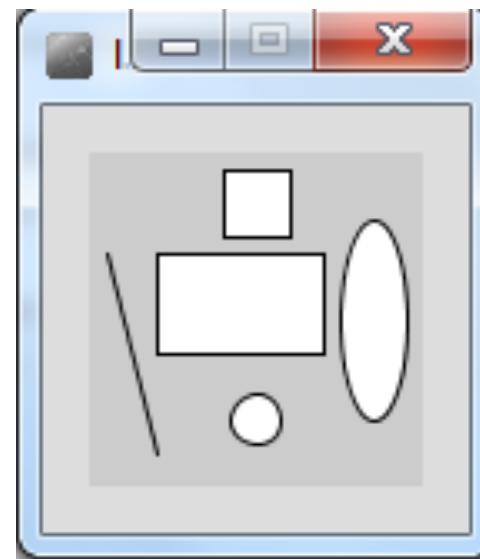
---

- Coordinate System in Computing.
- Functions in Processing.
- Basic Shapes.
- Formatting the display window.
- Syntax Errors.
- Logic Errors.

# 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()

The image shows the Processing 2.2.1 software interface. The top window is titled "sketch\_150615a | Processing 2.2.1". The menu bar includes File, Edit, Sketch, Tools, and Help. The toolbar contains icons for play, stop, file operations, and zoom. The code editor window shows the sketch name "sketch\_150615a" and the command "size(400,300);". Below this, there is sample code: "rect(20,30,50,30);", "rect(40,5,20,20);", "line(5,30,20,90);", "ellipse(85,50,20,60);", and "ellipse(50,80,15,15);". To the right is the preview window titled "sketch\_150615a" which displays a gray rectangle with a white border containing several geometric shapes: a small square at the top, a large rectangle in the center, a horizontal oval to the right, and a small circle below the rectangle. A blue rectangular box highlights the word "fullScreen();" at the bottom left.

```
sketch_150615a | Processing 2.2.1
File Edit Sketch Tools Help
Java
sketch_150615a
size(400,300);

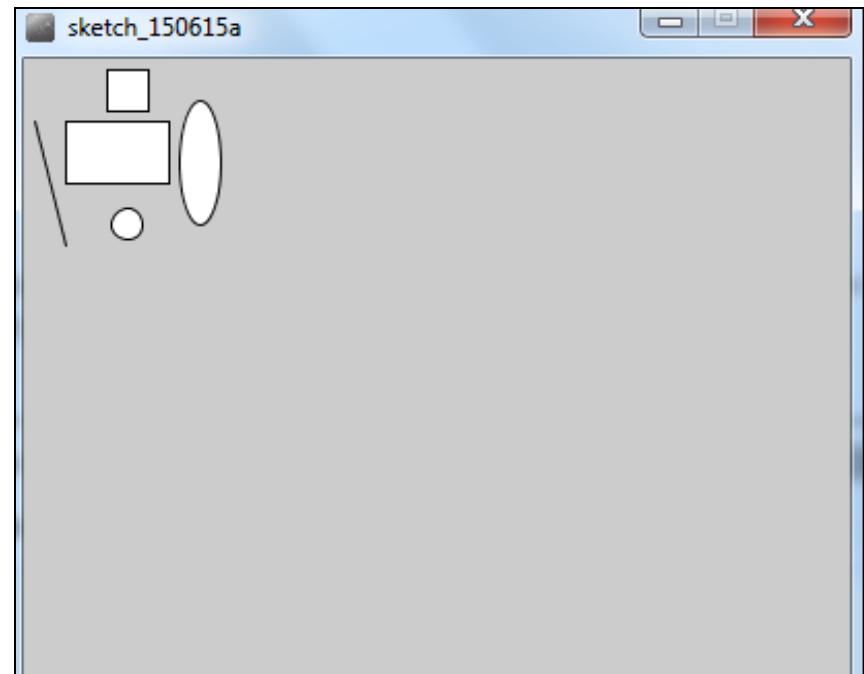
rect(20,30,50,30);
rect(40,5,20,20);
line(5,30,20,90);
ellipse(85,50,20,60);
ellipse(50,80,15,15);

fullScreen();
```

# Formatting the Display Window

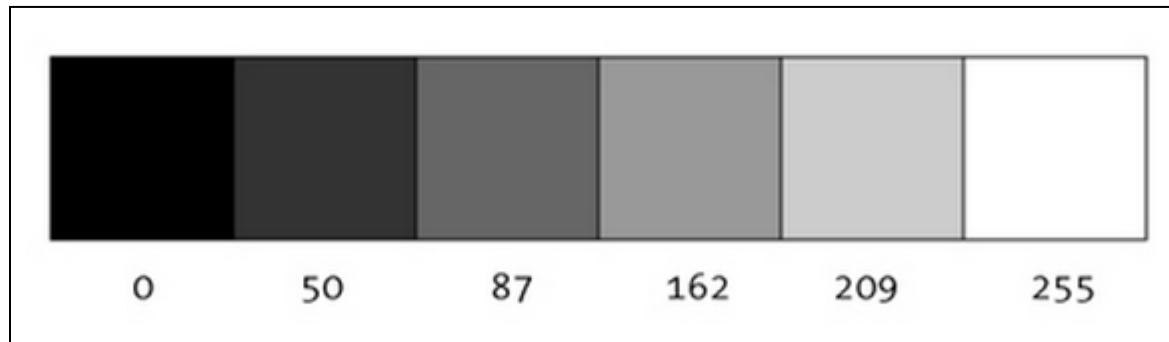
---

- Our display window looks less cramped now.
- But the default gray colour is not very appealing.
- We could use the **background** function to set the colour to something nicer.



# A Note on Colour First...Grayscale

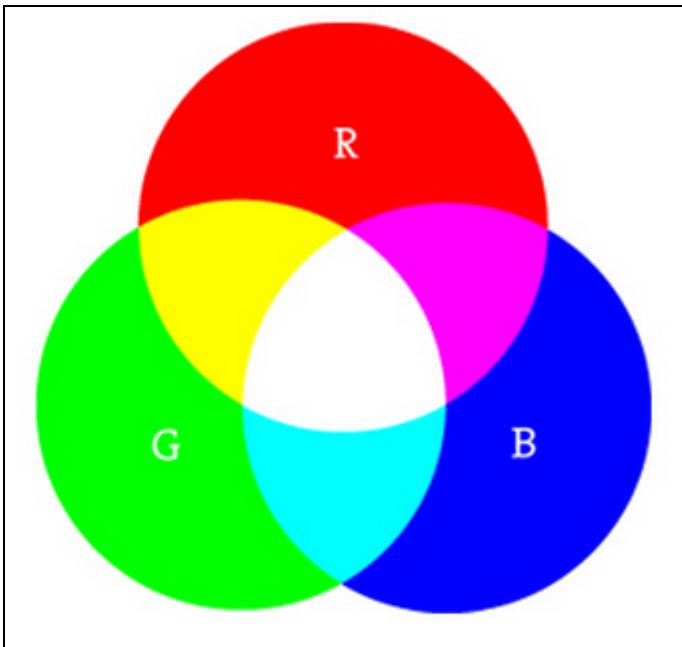
---



“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.”

# A Note on Colour First...RGB

---

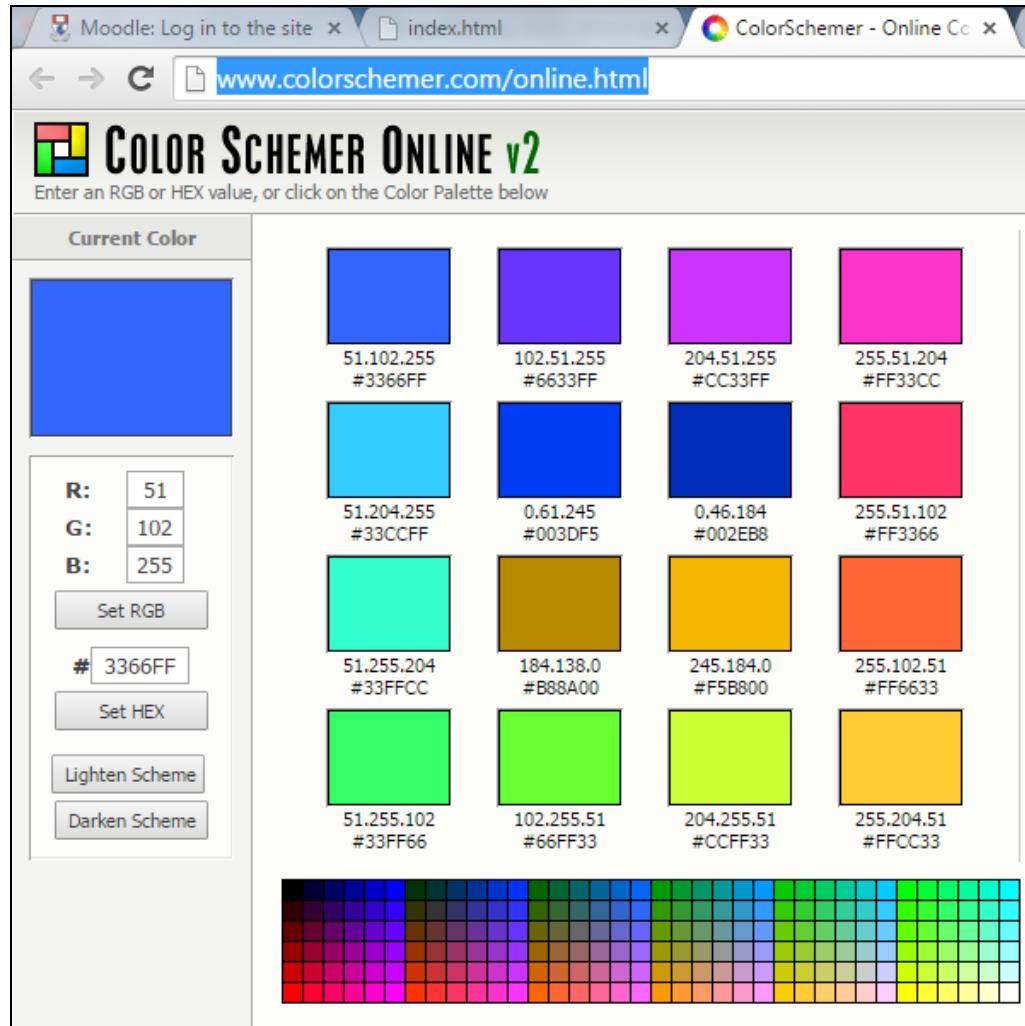


“As with grayscale, the individual color elements are expressed as ranges from 0 (none of that color) to 255 (as much as possible), and they are listed in the order R, G, and B.”

Digital colours are made by mixing the three primary colours of light (red, green, and blue).

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

# A Note on Colour First...RGB



<http://www.colorschemer.com/online.html>

# background() - Syntax

---

## background(grayscale)

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

## background(r, g, b)

r = red colour (a number between 0 and 255 inclusive)

g = green colour (a number between 0 and 255 inclusive)

b = blue colour (a number between 0 and 255 inclusive)

# background()

The image shows the Processing 2.2.1 software interface. The title bar reads "P sketch\_150615a | Processing 2.2.1". The menu bar includes File, Edit, Sketch, Tools, and Help. Below the menu is a toolbar with icons for play, stop, file operations, and run. The sketch name "sketch\_150615a" is displayed. A dropdown menu shows "Java". The code editor contains the following Pseudocode:

```
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);
```

To the right, the preview window titled "sketch\_150615a" displays a white canvas with several black shapes: a small rectangle at (20, 30), a larger rectangle at (40, 5), a line from (5, 30) to (20, 90), an ellipse at (85, 50) with dimensions 20x60, and another ellipse at (50, 80) with dimensions 15x15.

# background()

The image shows the Processing 2.2.1 software interface. On the left is the code editor window with the title "sketch\_150615a | Processing 2.2.1". The code is as follows:

```
size(400,300);
background(190,240,245);

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);
```

On the right is the preview window titled "sketch\_150615a" showing the output of the code. The background is light blue. It contains several shapes: a small white rectangle at the top left, a larger white rectangle below it, a thin black line from (5,30) to (20,90), a large light blue ellipse centered at (85,50) with dimensions 20x60, and a small light blue circle centered at (50,80) with dimensions 15x15.

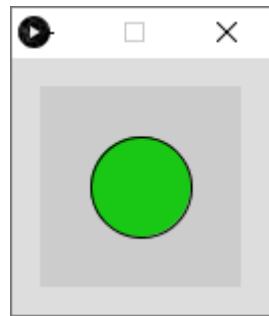
# Order

---

- Order of drawing is important

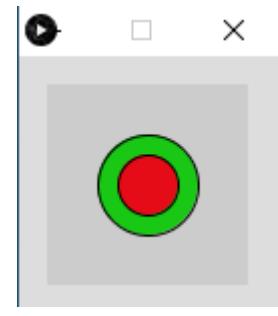
sketch\_191205a | Processing 3.5.3

```
File Edit Sketch Debug Tools Help  
sketch_191205a ▾  
1  
2 fill(230, 12, 23);  
3 ellipse(50, 50, 30, 30);  
4  
5 fill(25, 200, 20);  
6 ellipse(50, 50, 50, 50);  
7  
8  
9  
10  
11  
12  
13  
14
```



sketch\_191205a | Processing 3.5.3

```
File Edit Sketch Debug Tools Help  
sketch_191205a ▾  
1 fill(25, 200, 20);  
2 ellipse(50, 50, 50, 50);  
3  
4 fill(230, 12, 23);  
5 ellipse(50, 50, 30, 30);  
6  
7  
8  
9  
10  
11
```



# Topics List

---

- Coordinate System in Computing.
- Functions in Processing.
- Basic Shapes.
- Formatting the display window.
- Syntax Errors.
- Logic Errors.

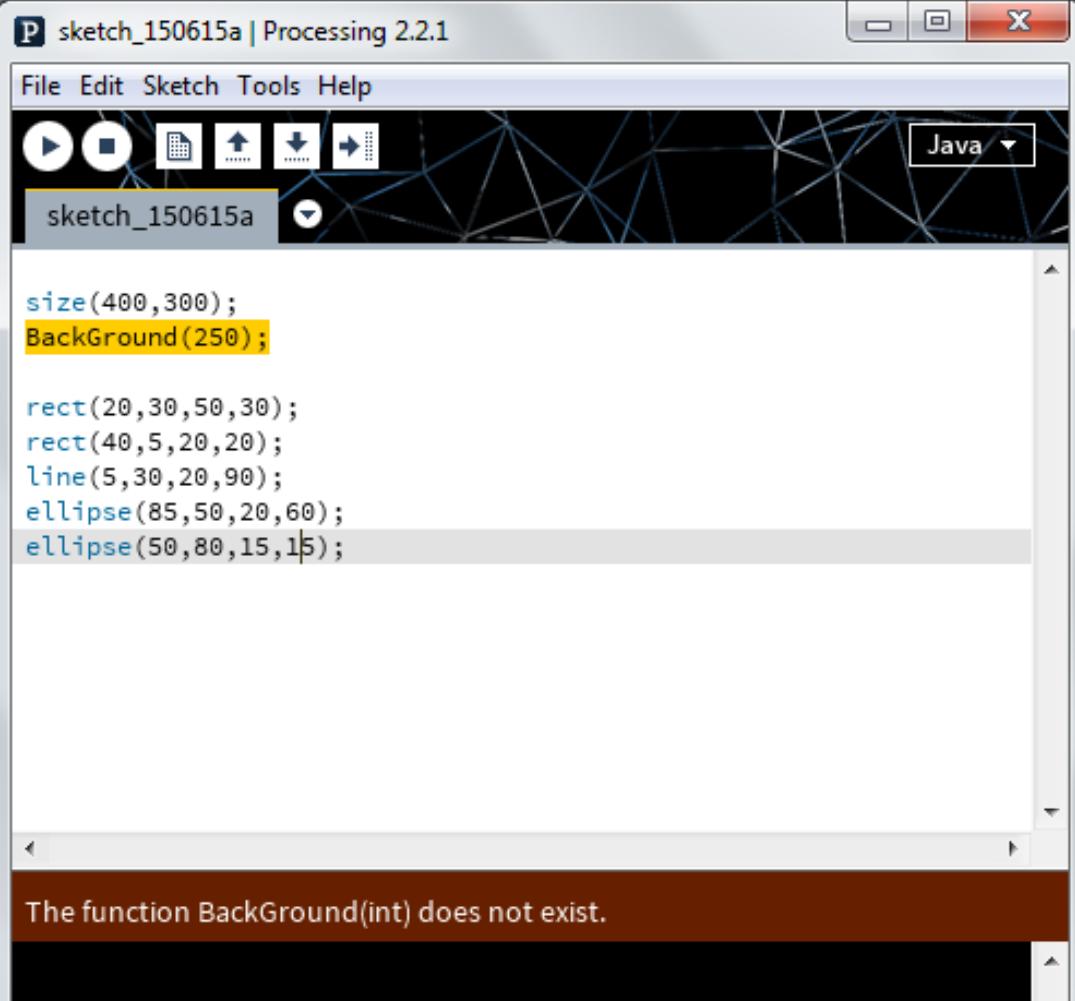
# Syntax and Syntax Errors

---

- You will have seen the term **Syntax** mentioned above.
- Syntax are the rules you must follow when writing well-formed statements in a programming language.
- When you don't follow the rules, Processing will not run your code; instead you will get an error.
- Some syntax error examples are on the upcoming slides.

# Syntax Errors

*The spelling of  
the background  
function must be  
identical to the  
spelling below  
(case sensitive!).*



A screenshot of the Processing 2.2.1 software interface. The title bar says "sketch\_150615a | Processing 2.2.1". The menu bar includes File, Edit, Sketch, Tools, and Help. Below the menu is a toolbar with various icons. The sketch window shows the following code:

```
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);
```

The word "BackGround" is highlighted in yellow, indicating a syntax error. A red error message at the bottom of the sketch window reads: "The function BackGround(int) does not exist."

**background(r, g, b)**

r = red colour (a number between 0 and 255 inclusive)

g = green colour (a number between 0 and 255 inclusive)

b = blue colour (a number between 0 and 255 inclusive)

# Syntax Errors

*The background function has too many arguments passed to it i.e.*

- *RGB version is defined with 3 parameters.*
- *Grayscale version is defined with 1 parameter.*

The screenshot shows the Processing 2.2.1 IDE interface. The title bar reads "sketch\_150615a | Processing 2.2.1". The menu bar includes File, Edit, Sketch, Tools, and Help. Below the menu is a toolbar with various icons. The code editor window contains the following sketch code:

```
size(400,300);
background(250,250,343,232,456);

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);
```

A tooltip at the bottom of the code editor window displays the error message: "The method background(int, float) in the type PApplet is not applicable for the arguments (int, int, int, int, int)".

**background(r, g, b)**

r = red colour (a number between 0 and 255 inclusive)

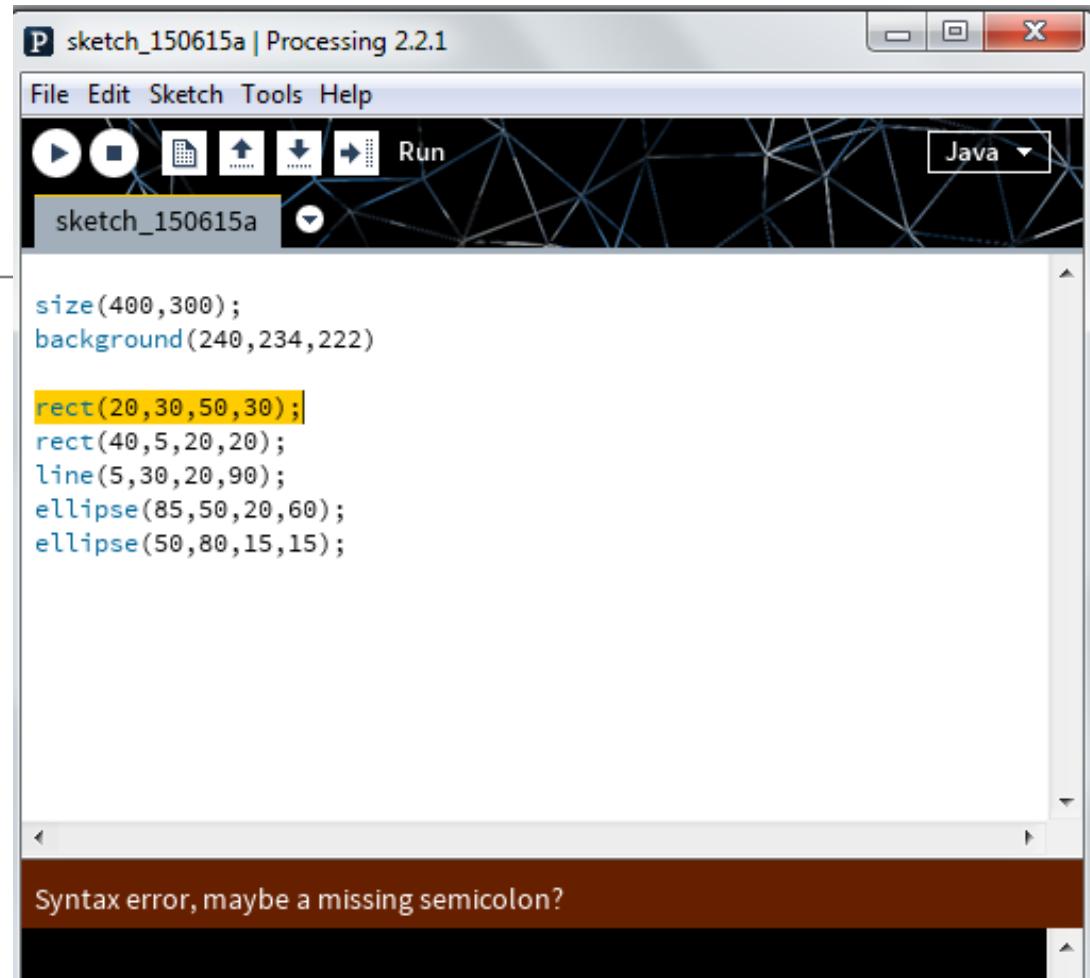
g = green colour (a number between 0 and 255 inclusive)

b = blue colour (a number between 0 and 255 inclusive)

# Syntax Errors

*The semi-colon (;) is missing at the end of the statement.*

*Java needs a statement terminator for each line!*



The screenshot shows the Processing 2.2.1 IDE interface. The title bar reads "sketch\_150615a | Processing 2.2.1". The menu bar includes File, Edit, Sketch, Tools, and Help. Below the menu is a toolbar with icons for play, stop, save, and run, followed by a "Run" button. A dropdown menu shows "Java". The code editor window contains the following Java code:

```
size(400,300);
background(240,234,222)

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);
```

A red horizontal bar at the bottom of the code editor displays the message "Syntax error, maybe a missing semicolon?".

**background(r, g, b)**

r = red colour (a number between 0 and 255 inclusive)

g = green colour (a number between 0 and 255 inclusive)

b = blue colour (a number between 0 and 255 inclusive)

# Topics List

---

- Coordinate System in Computing.
- Functions in Processing.
- Basic Shapes.
- Formatting the display window.
- Syntax Errors.
- Logic Errors.

# Logic Errors

---

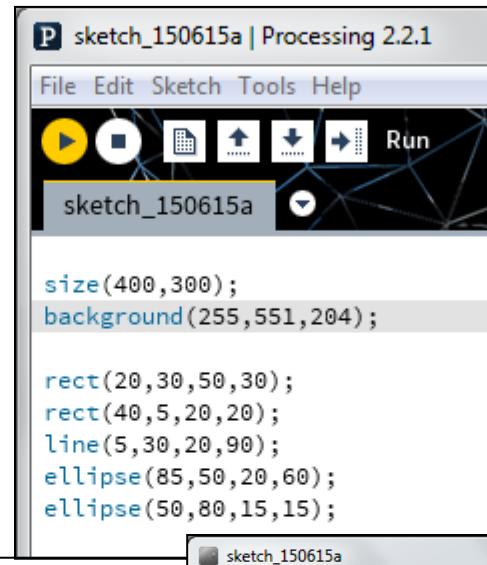
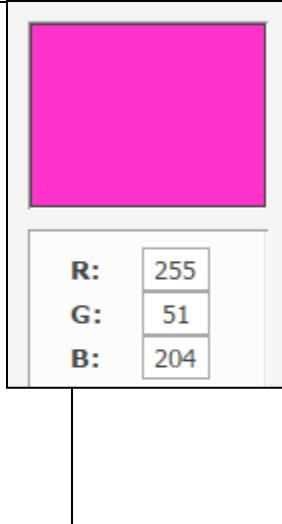
In computer programming, a **logic error** is a bug in a program that causes it to operate incorrectly, but not to terminate abnormally (or crash). A **logic error** produces unintended or undesired output or other behaviour, although it may not immediately be recognised as such.

[Logic error - Wikipedia, the free encyclopedia](#)

[en.wikipedia.org/wiki/Logic\\_error](http://en.wikipedia.org/wiki/Logic_error)

# Logic Errors

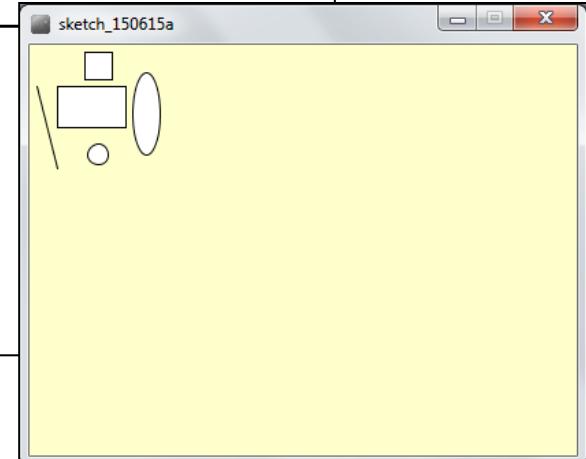
Say we wanted a pink background for our display window.



```
size(400,300);
background(255,551,204);

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);
```

- However, we incorrectly enter the **G** colour as 551 instead of 51.
- We now have a yellowish background.
- This is an example of a simple logic error.



# Questions?

---





Except where otherwise noted, this content is licensed under a Creative Commons Attribution-NonCommercial 3.0 License.

For more information, please see <http://creativecommons.org/licenses/by-nc/3.0/>

Produced  
by:  
Dr. Siobhán Drohan  
Mairead Meagher  
Sinéad Walsh



Waterford Institute of Technology  
INSTITIÚID TEICNEOLAÍOCHTA PHORT LÁIRGE

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