Crash Course 1: Introduction to Processing

ST. MARY'S HIGH SCHOOL

Welcome to Introduction to Processing!

Here’s what we’ll be covering:

- What is programming/coding?
- What is Processing?
- Pre-defined functions
  - rect()
  - line()
  - ellipse()
  - random()
  - fill()
  - print() and println()
- Summary

What is programming?

- Programming is a creative process which provides instructions for a computer. However computers don't understand the meaning of words like humans do - so how do we write something that a computer will be able to understand?

- Programming languages allow us to use *high level* commands that make sense to us, and with the use of a *compiler* convert these into *low level* commands (often a series of zeros and ones) that the computer can interpret

- There are multiple programming languages, such as Python, C#, C++, Java, and many more which all have their own advantages and disadvantages. Processing is one of these programming languages.

What is programming? (2)

- Due to the highly technological world we live in, programming is essentially in every aspect of our lives. From the phones we use, to the cars we drive - they all rely on some sort of programming

- Coding is the basis of fields such as software engineering and computer science, but can be found and applied in virtually all industries, especially in electronics, automation, app development, business, e-commerce, web design or even medicine

- It’s an extremely desirable and sought-after skill for employment and post-secondary education in today’s increasingly technology dependent society

What is Processing?

- Processing is a user-friendly, open source, Java-based programming language which is somewhat of a sketchbook, enabling you to code in a very visual way
- You can start your own programming adventure by downloading the Processing IDE (Integrated Development Environment)
- Link to download Processing: https://processing.org/download/
- Detailed reference for Processing: https://processing.org/reference/
- Note that most lines of code must end with a semicolon (;) to be compiled properly

Predefined functions

- A function is a small, modular, unit of code that can be “called” to complete a defined task
- Functions can be used repetitively
- They can be user-defined (created by you) or predefined (created by the makers of Processing)
- Predefined functions are free for you to use and are stored in a library included in your download of Processing
- For now, we will focus only on predefined functions

Predefined functions (2)

- The names of predefined functions are key words recognized by the compiler
- Some examples of predefined (or library) functions are rect(), size(), ellipse(), random(), print(), println(), fill(), stroke(), and line()
- Most functions require inputs or parameters to be put between the brackets to specify details that influence the function's output
- The size() function sets up the size of your sketching/display window and requires two parameters: the width and the height
- Typing in the IDE size(500,500); opens a 500 x 500 pixel window (go try it!)

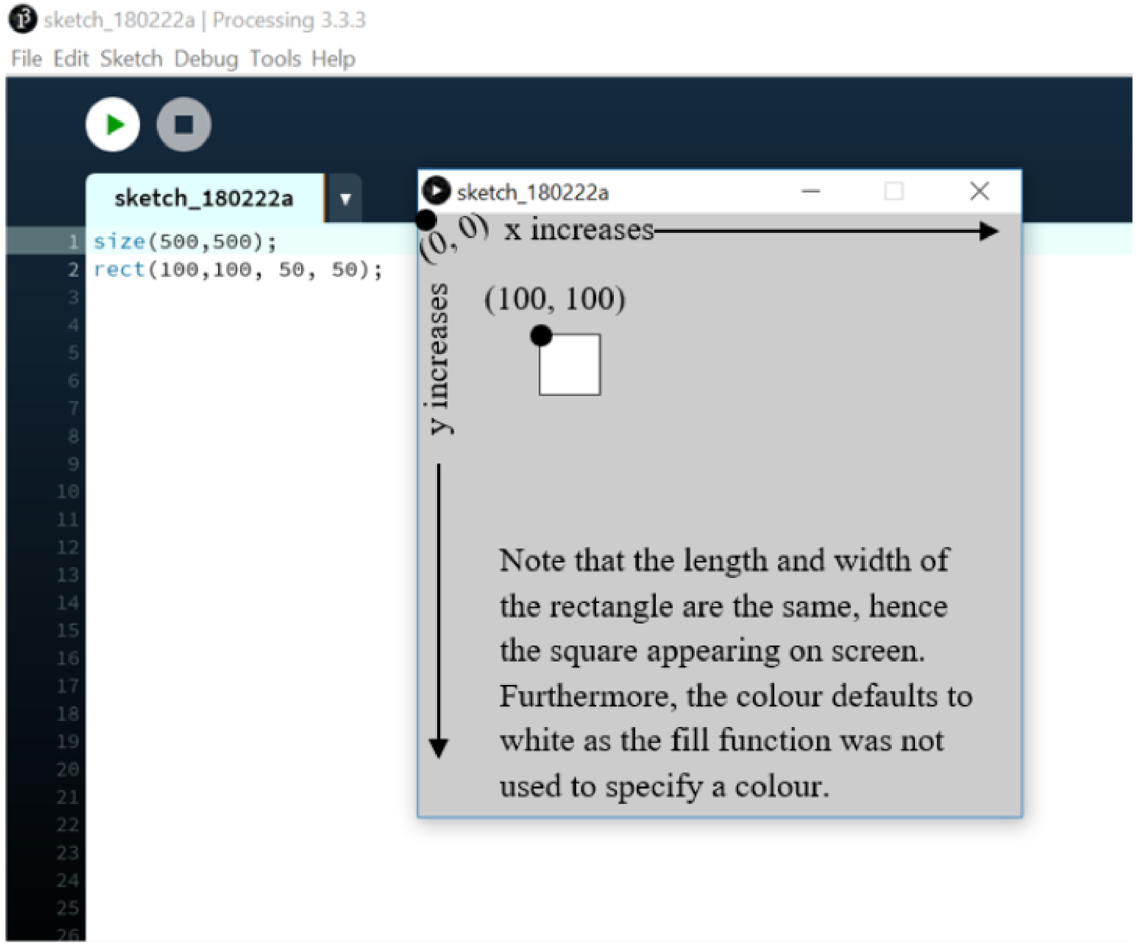
Predefined functions – rect()

- A function like rect() will require four parameters or inputs to draw a rectangle: an x-coordinate, a y-coordinate, width and height
- Syntax: rect(a, b, c, d);

Parameters		
	a	float: x-coordinate of the rectangle by default
	b	float: y-coordinate of the rectangle by default
	c	float: width of the rectangle by default
	d	float: height of the rectangle by default

- For instance, rect(100, 100, 50, 50);, when compiled, draws a square with width and height equal to 50 units, positioned at (100, 100) on the sketching window
- Note: syntax is just a fancy word for “grammar”, the correct way to write code so that the compiler (what translates your code into 0s and 1s) can understand it

Predefined functions – rect()


sketch_180222a | Processing 3.3.3
File Edit Sketch Debug Tools Help
sketch_180222a
sketch_180222a
(0, 0)
x increases
y increases
(100, 100)

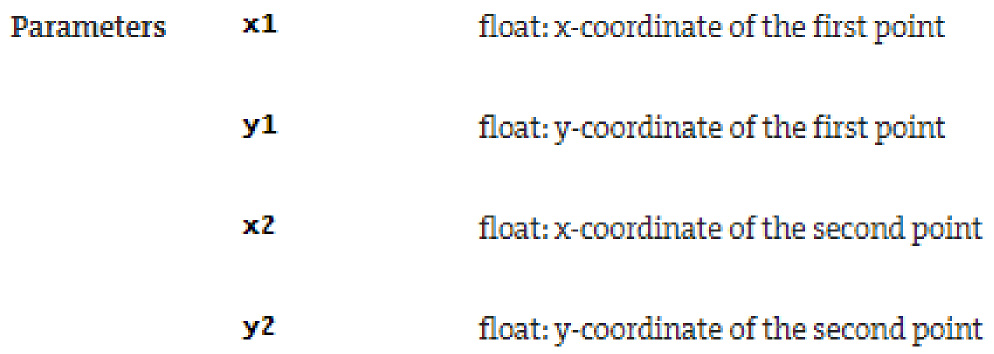
```
size(500,500);
rect(100,100, 50, 50);
```

Note that the length and width of the rectangle are the same, hence the square appearing on screen. Furthermore, the colour defaults to white as the fill function was not used to specify a colour.

Note that on the Processing sketching window, x values increase from left to right but unlike a normal Cartesian graph, y -values increase from top to bottom

Predefined functions – line()

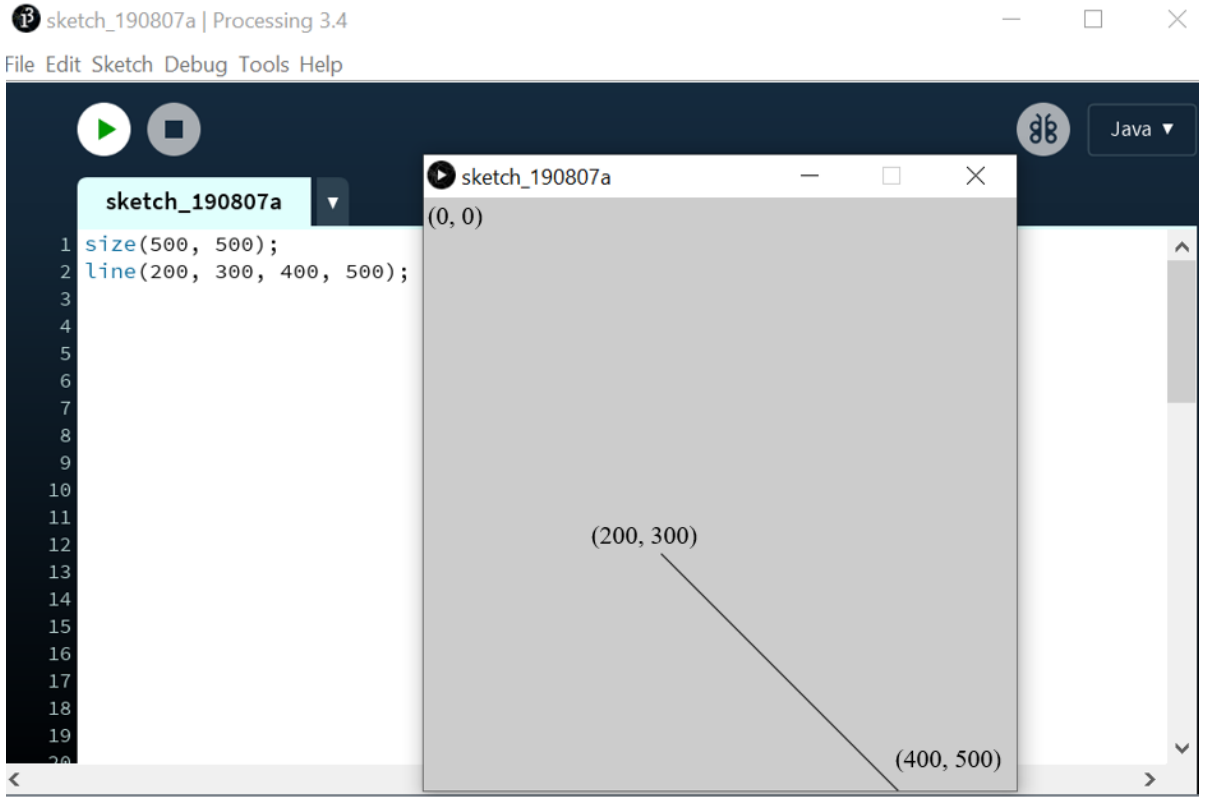
- The line() function also requires four parameters: the x and y positions of an initial point as well as the x and y positions of a terminal point
- Syntax: line(x1, y1, x2, y2);



Parameters	x1	float: x-coordinate of the first point
	y1	float: y-coordinate of the first point
	x2	float: x-coordinate of the second point
	y2	float: y-coordinate of the second point

Predefined functions – line()

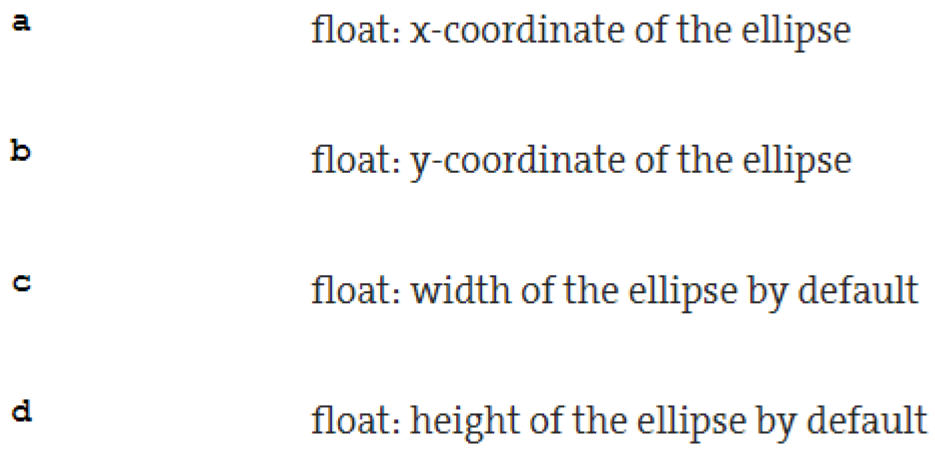
- For example, line(200, 300, 400, 500); generates a line connecting the points (200, 300) and (400, 500)


sketch_190807a | Processing 3.4
File Edit Sketch Debug Tools Help
Java
sketch_190807a
(0, 0)
(200, 300)
(400, 500)

```
size(500, 500);
line(200, 300, 400, 500);
```

Predefined functions – ellipse()

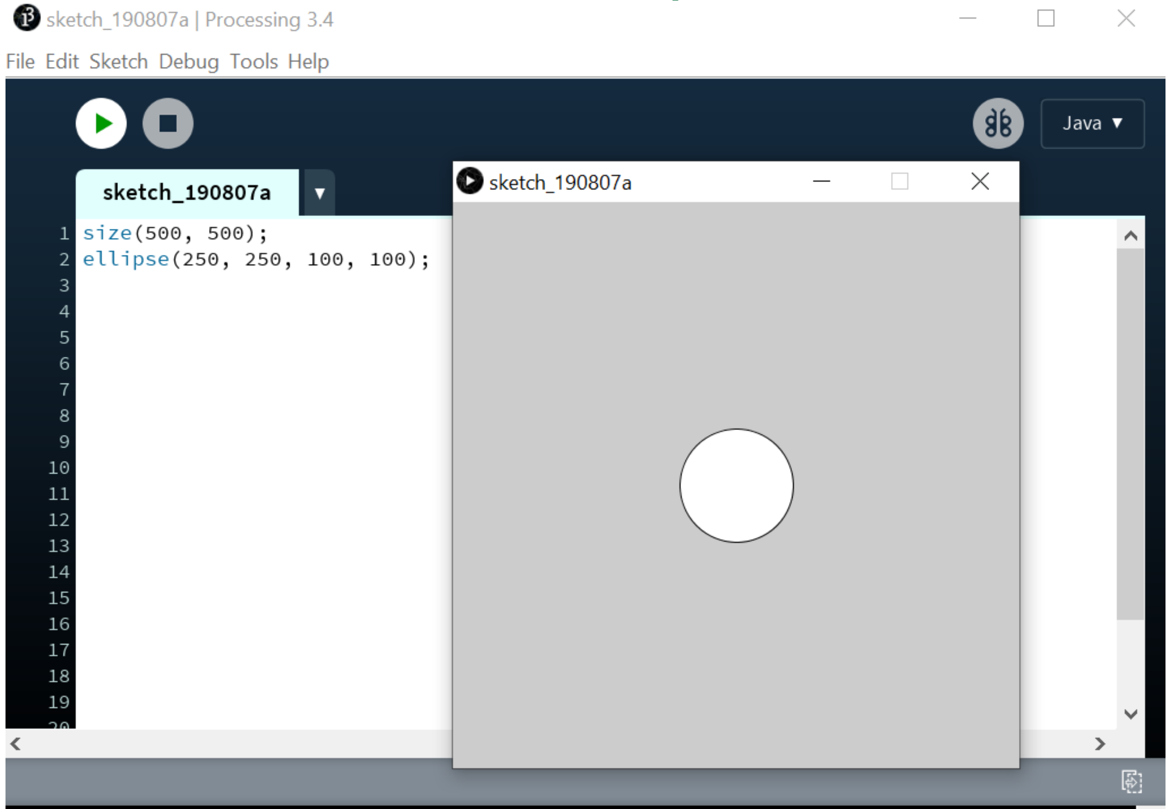
- The ellipse() function in Processing usually requires four parameters: the x and y coordinates of its centre, as well as its width and height
- If the width and height are equal, you have a circle
- Syntax: ellipse(a, b, c, d);



a	float: x-coordinate of the ellipse
b	float: y-coordinate of the ellipse
c	float: width of the ellipse by default
d	float: height of the ellipse by default

Predefined functions – ellipse()

- For example, ellipse(250, 250, 100, 100); creates a circle at (250, 250) with a height and width of 100 units (a diameter of 100 pixels)


sketch_190807a | Processing 3.4
File Edit Sketch Debug Tools Help
Java
sketch_190807a
sketch_190807a

```
size(500, 500);
ellipse(250, 250, 100, 100);
```

Predefined functions – random()

- random() outputs a random number
- For instance, random(15) will return a random number from 0 up to but not including 15
- random(5.7, 15) will return a random number from 5.7 up to but not not including 15
- Random is an example of a function that has optional parameters - not all parameters have to be filled in, as they have some default value
- Syntax: random(high); or random(low, high);

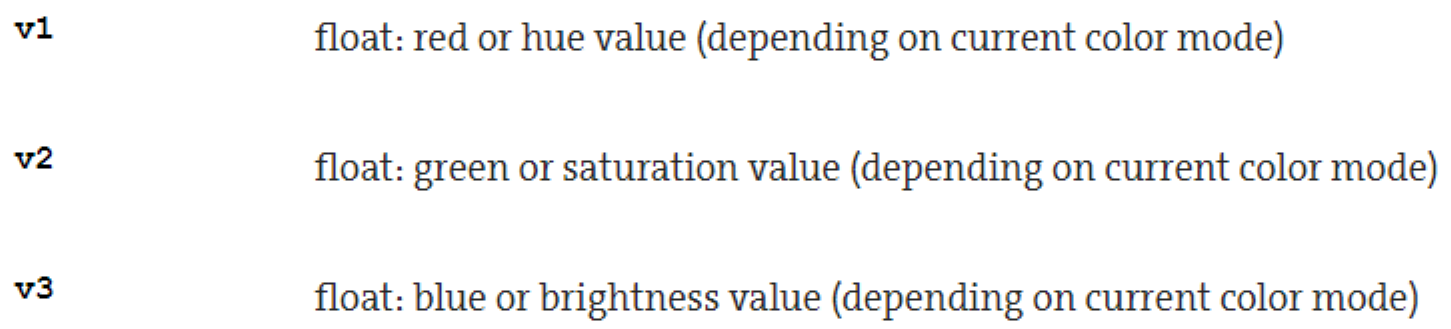
`low`	float: lower limit
`high`	float: upper limit

Predefined functions – fill()

- The fill() function allows you to colour shapes according to the RGB (Red, Green, Blue) colour scale with values from 0 to 255
- The fill() function takes three parameters
- fill(0, 0, 0); fills with black
- fill(255, 255, 255); fills with white
- Any other colour is generated by having various numbers in the first position (amount of red), in the second position (amount of green) and in the third position (amount of blue)

Predefined functions – fill()

- Syntax: fill(v1, v2, v3);

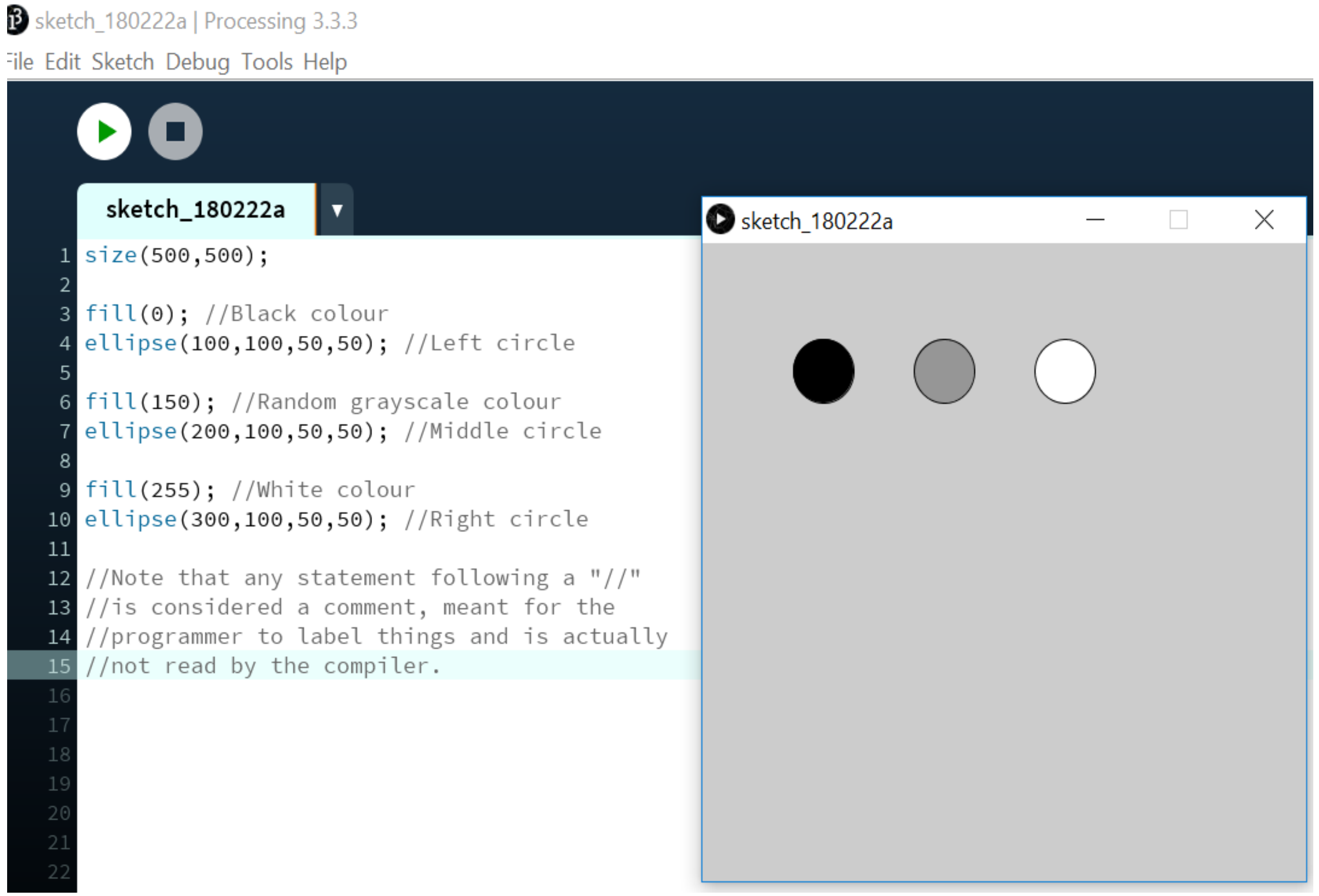


`v1`	float: red or hue value (depending on current color mode)
`v2`	float: green or saturation value (depending on current color mode)
`v3`	float: blue or brightness value (depending on current color mode)

Predefined functions – fill()

- For example, when you write fill(255, 0, 0);, you have turned on all the red light R but have 0 for the amounts of green and blue light G and B (hence you see red)
- Note that the colour black may be achieved with a single parameter fill(0), because it assumes the other parameters to be the same and similarly the colour white may be obtained with fill(255)
- As you might suspect, any value in between 0 and 255 would generate some other colour on the grayscale
- stroke() works exactly like fill() but is used to colour lines instead of shapes

Predefined functions – fill()


sketch_180222a | Processing 3.3.3
File Edit Sketch Debug Tools Help
sketch_180222a
sketch_180222a

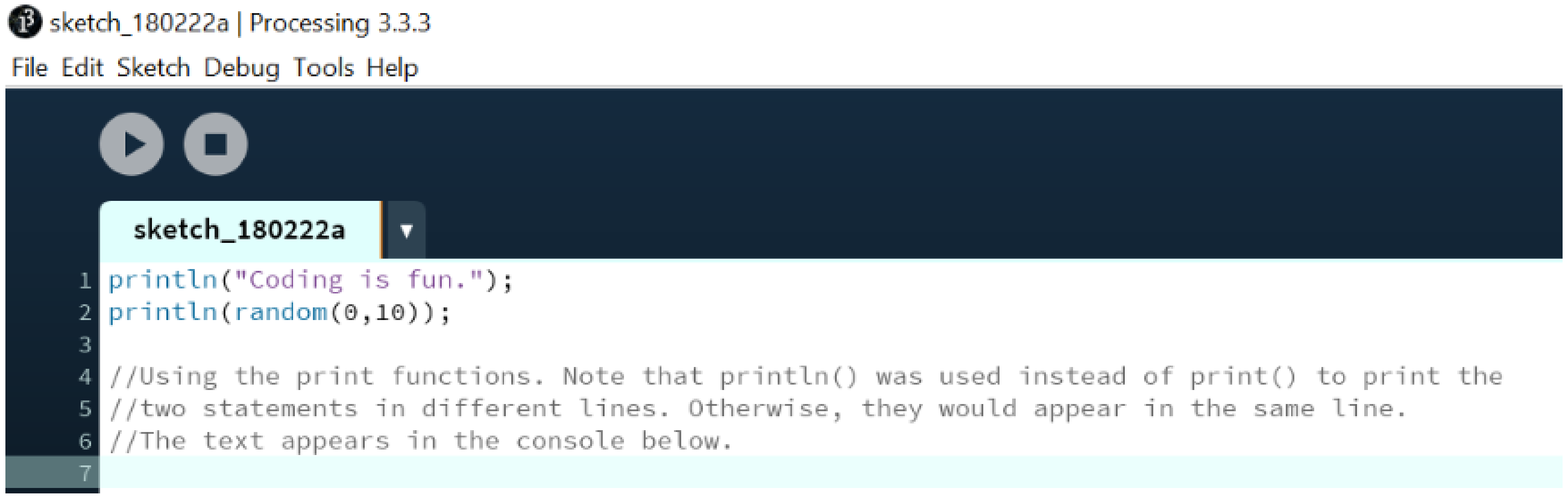
```
size(500,500);

fill(0); //Black colour
ellipse(100,100,50,50); //Left circle

fill(150); //Random grayscale colour
ellipse(200,100,50,50); //Middle circle

fill(255); //White colour
ellipse(300,100,50,50); //Right circle

//Note that any statement following a "//"
//is considered a comment, meant for the
//programmer to label things and is actually
//not read by the compiler.
```

Predefined functions – print() and println()

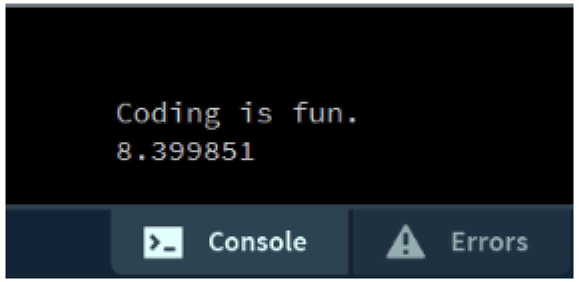
- The print() function prints text or function outputs to the *console* (which is the black box underneath your code)
- To print a message, you must have double quotations, i.e. print(“Coding is fun.”)
- The print() function prints in the same line every time it is called
- The println() function is similar to the print() function but it prints in a new line every time it gets called (similar to pressing the “Enter” key on a keyboard before adding more text)
- Syntax: print(what) or println(what)

`what`	Object, String, float, char, boolean, or byte: data to print to console

Predefined functions – print() and println()


sketch_180222a | Processing 3.3.3
File Edit Sketch Debug Tools Help
sketch_180222a

```
println("Coding is fun.");
println(random(0,10));

//Using the print functions. Note that println() was used instead of print() to print the
//two statements in different lines. Otherwise, they would appear in the same line.
//The text appears in the console below.
```



```
Coding is fun.
8.399851
```

Don't forget that the Processing reference page linked on Slide 5 contains all the information you need for each predefined function!

Summary

- Well would you look at that, you’re coding! We’re only going up from here! Now let’s review what we’ve learned.
- In this crash course, we learned what coding is all about and were introduced to Processing
- Computer programming consists of writing commands in source code that get compiled into machine code (0s and 1s) for execution
- Coding can be found in many industries and is consequently a very desirable skill to have
- Processing is an open source, Java-based language with an integrated development environment (IDE) for you to write your code in, which helps people learn to code in a visual way

Summary (2)

- Processing offers several predefined or library functions such as rect(), line(), ellipse(), random(), fill(), print(), and println() that are ready to complete basic tasks for you
- Functions are small chunks of code that can be used repetitively
- Documentation for predefined functions covered and many more can be found on the Processing website at https://processing.org/reference/
- You have completed: **1. Introduction to Processing**
- Up next: **2. Variables and Data Types**