

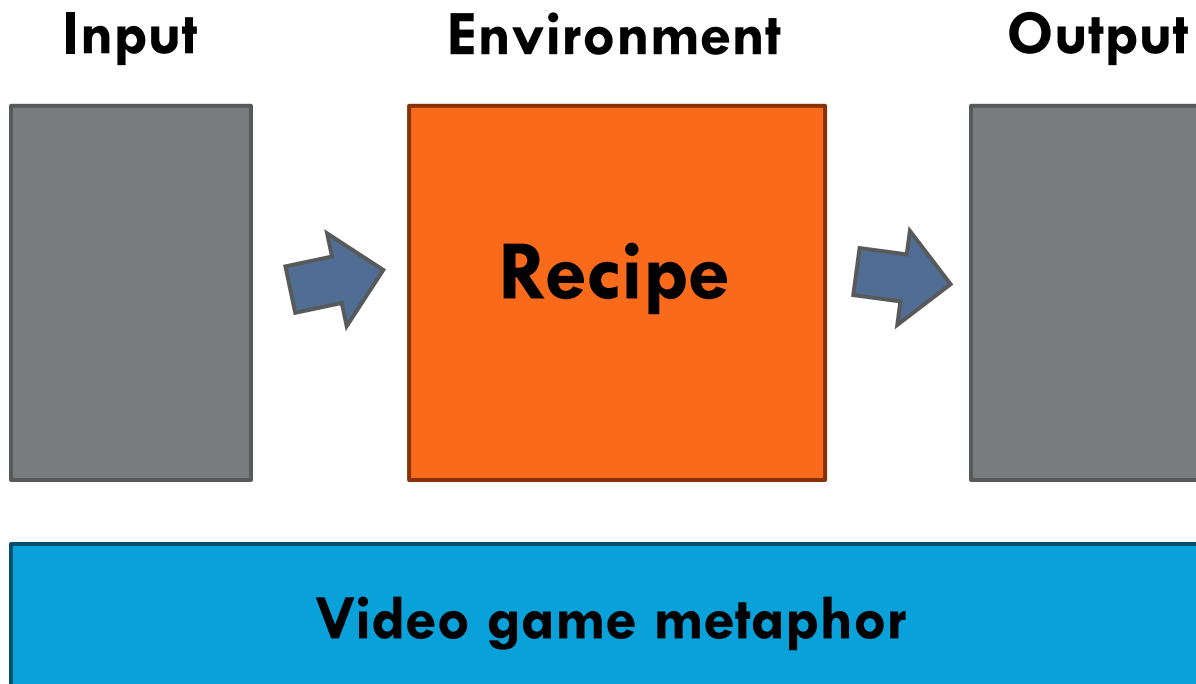
INTERACTIVE MULTIMEDIA DESIGN

With an AI touch :)

Interactive Multimedia Design

2

- It's important to separate the inputs and outputs



Computer programming

Computer programming

4



Computer programming

5



Computer programming

6



Computer programming

7

- A lot of programming languages
 - ▣ C, C++, Java, Perl, Python, JavaScript, PHP, Ruby, ...
- A lot of terminology
 - ▣ Variable, value, type, class, function, method, routine, interface, reference, array, conditional, loop, ...
- A lot of packages used on top of core languages
 - ▣ Rails, Django, JQuery, OpenCV, ...

Computer programming

8

- The fundamental principles are **simple** and **similar** to all programming languages
- You can use a lot of functionalities as “**Lego-bricks**” as long as you understand how to put them together
- **Coding is your friend!**

Computer programming

9

- *“The programmers of tomorrow are the wizards of the future. You’re gonna look like you have magic powers”*
Gabe Newell, founder of Valve



Computer programming

10

- *“Coding is the closest thing we have to a superpower”*
Drew Houston, creator of Dropbox



Computer programming

11

- We will look over some basic things to get you started with programming
- There is an **enormous** amount of information **online**, there is always someone who had a similar challenge, and usually there is documentation for it
- It is assumed that you have no previous knowledge or experience of programming

Computer programming

12

- We will look over some basic things to get you started with programming
- There is an **enormous** amount of information **online**, there is always someone who had a similar challenge, and usually there is documentation for it
- It is assumed that you have no previous knowledge or experience of programming
 - ▣ For those of you who have, be patient :-)

Computer programming

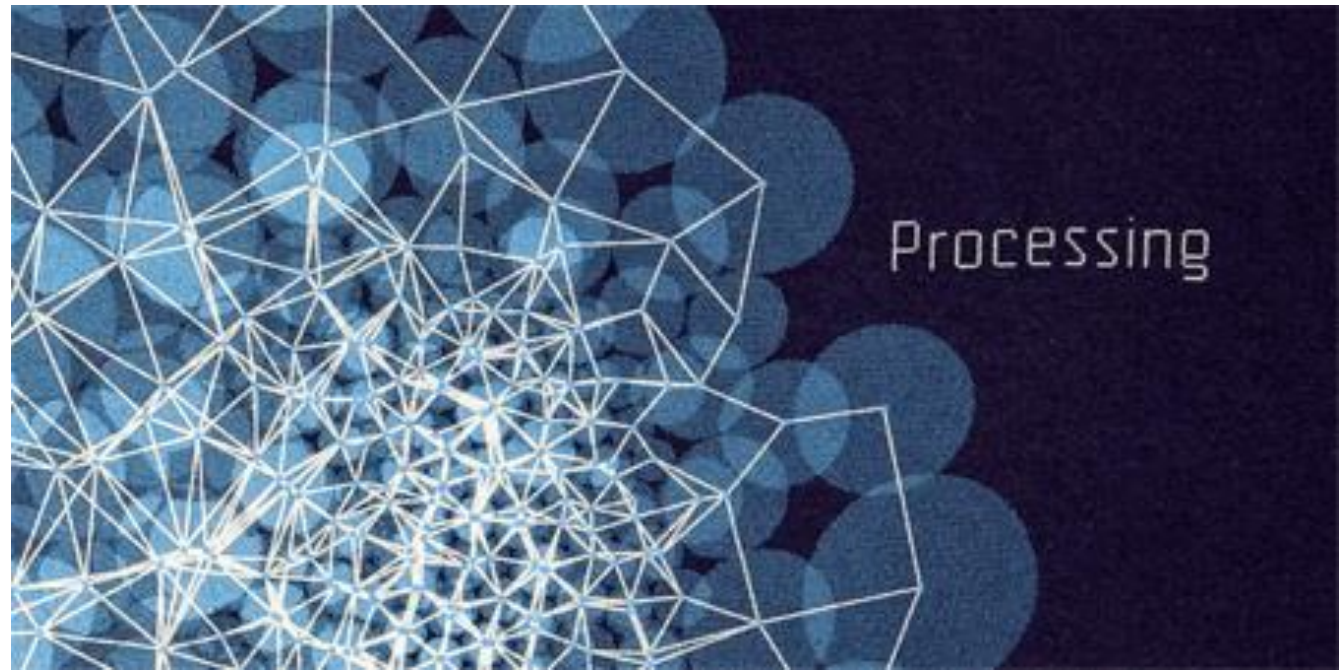
13

- We will look over some basic things to get you started with programming
- There is an **enormous** amount of information **online**, there is always someone who had a similar challenge, and usually there is documentation for it
- It is assumed that you have no previous knowledge or experience of programming
 - ▣ For those of you who have, be patient :-)

Processing programming language

16

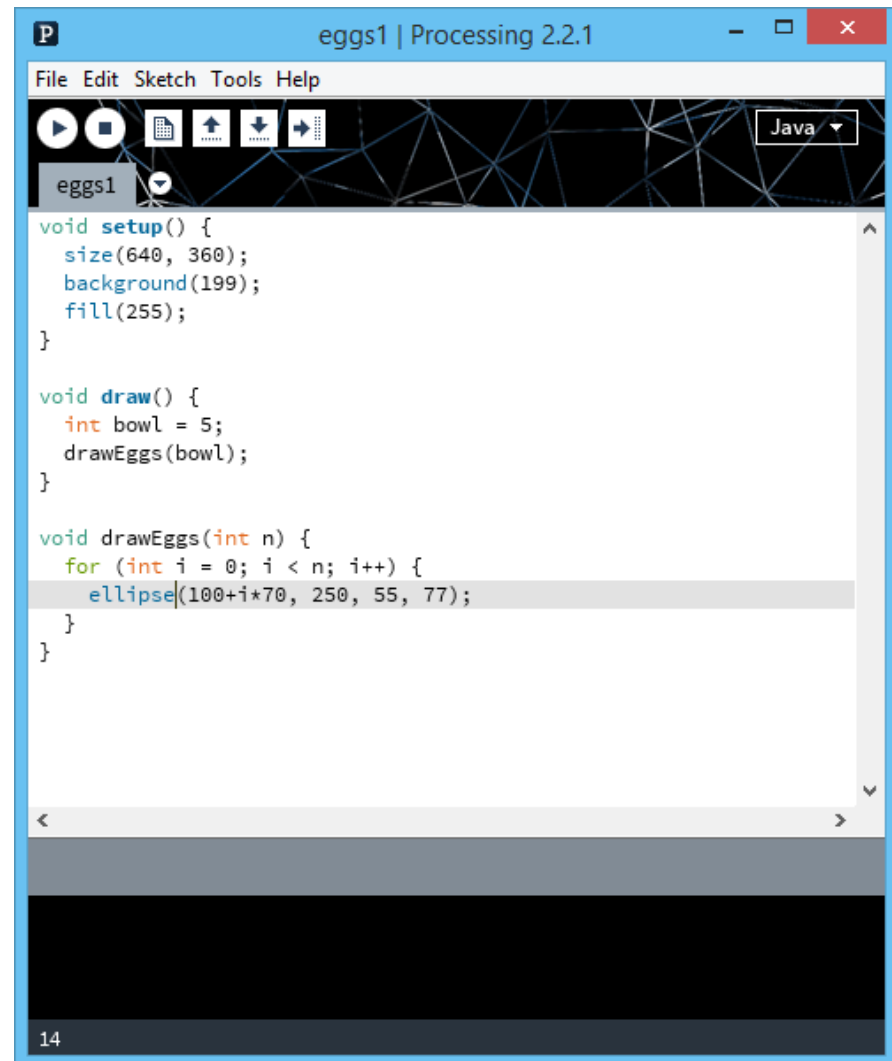
- Download the Processing language and programming environment from the following link
 - <https://processing.org/download/>



Magic recipes, Example 1

17

- Download file eggs1.zip from the following link:
 - <http://tinyurl.com/int-mult-2015-pde>
- Unzip the file and open eggs1.pde from folder eggs1
- What you see is the source code of a simple program



The screenshot shows the Processing IDE window titled 'eggs1 | Processing 2.2.1'. The menu bar includes 'File', 'Edit', 'Sketch', 'Tools', and 'Help'. The toolbar contains icons for play, stop, refresh, and other sketch controls. A dropdown menu shows 'Java'. The sketch name 'eggs1' is visible in the top left of the editor area. The code is as follows:

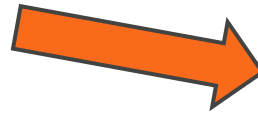
```
void setup() {  
  size(640, 360);  
  background(199);  
  fill(255);  
}  
  
void draw() {  
  int bowl = 5;  
  drawEggs(bowl);  
}  
  
void drawEggs(int n) {  
  for (int i = 0; i < n; i++) {  
    ellipse(100+i*70, 250, 55, 77);  
  }  
}
```

The number '14' is visible in the bottom left corner of the IDE window.

Magic recipes, Example 1

18

- Let's focus on the middle part of this simple program



```
eggs1 | Processing 2.2.1
File Edit Sketch Tools Help
Java
eggs1
void setup() {
  size(640, 360);
  background(199);
  fill(255);
}
void draw() {
  int bowl = 5;
  drawEggs(bowl);
}
void drawEggs(int n) {
  for (int i = 0; i < n; i++) {
    ellipse(100+i*70, 250, 55, 77);
  }
}
```

14

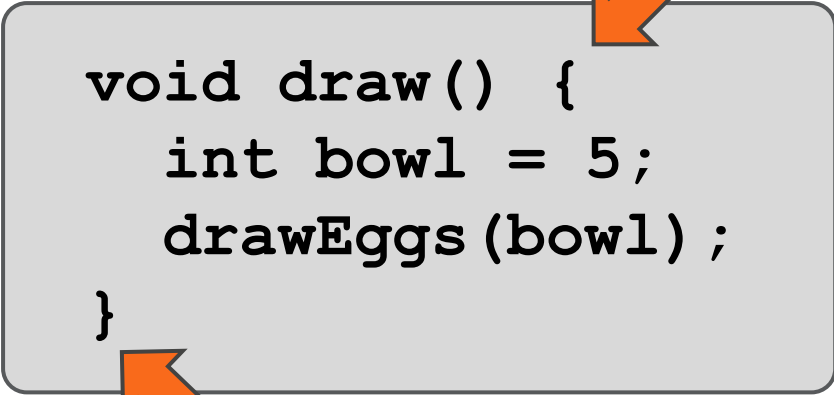
Magic recipes, Example 1

19

```
void draw() {  
    int bowl = 5;  
    drawEggs(bowl);  
}
```

Magic recipes, Example 1

20



```
void draw() {  
    int bowl = 5;  
    drawEggs (bowl) ;  
}
```

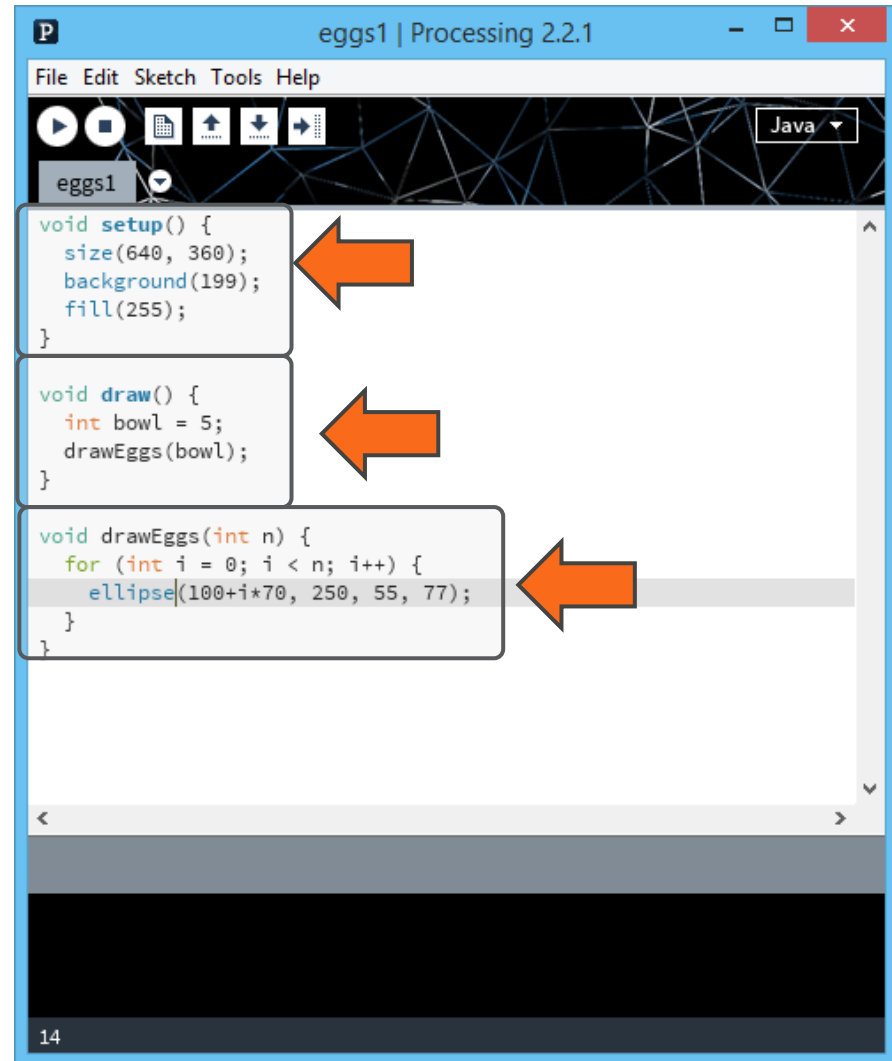
The code block is enclosed in a light gray rounded rectangle. Two orange arrows point to the opening curly brace '{' at the top right and the closing curly brace '}' at the bottom left, highlighting the brackets used to define the code block.

- In computer programming we separate **blocks** of code using **brackets**
- This is the block of code that tells to the system what to draw on our screen when we run it

Magic recipes, Example 1

21

- There are three blocks in this simple program
- We use indentation to make it easy to see where blocks start/end
- Use Control-T or Command-T to auto format the code

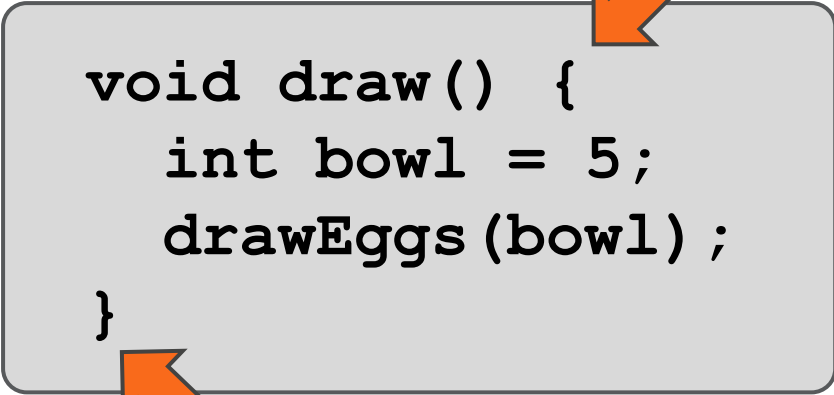


```
void setup() {  
  size(640, 360);  
  background(199);  
  fill(255);  
}  
  
void draw() {  
  int bowl = 5;  
  drawEggs(bowl);  
}  
  
void drawEggs(int n) {  
  for (int i = 0; i < n; i++) {  
    ellipse(100+i*70, 250, 55, 77);  
  }  
}
```

14

Magic recipes, Example 1

22



```
void draw() {  
    int bowl = 5;  
    drawEggs (bowl) ;  
}
```

The code block is enclosed in a light gray rounded rectangle. Two orange arrows point to the opening curly brace '{' at the top right and the closing curly brace '}' at the bottom left, highlighting the code's boundaries.

- In computer programming we separate **blocks** of code using **brackets**
- This is the block of code that tells to the system what to draw on our screen when we run it

Magic recipes, Example 1

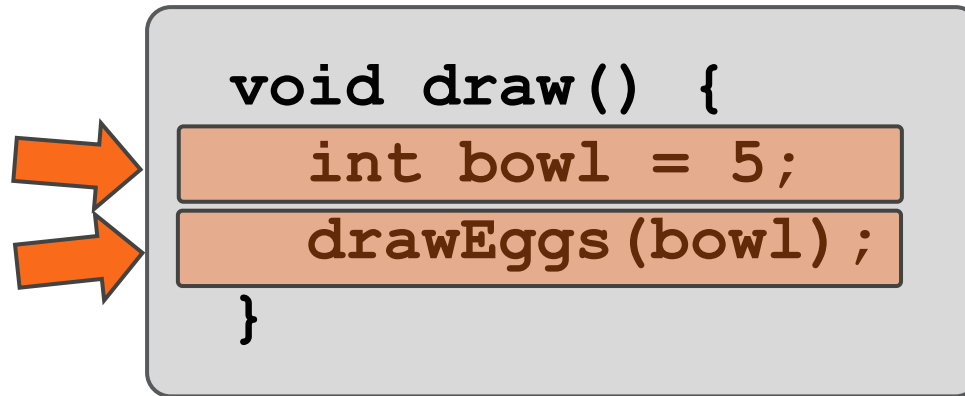
23

```
void draw() {  
    int bowl = 5;  
    drawEggs(bowl);  
}
```

- This **block** has a name: “void draw()”

Magic recipes, Example 1

24



- This **block** has a name: “`void draw()`”
- And it consists of two **statements**, each one on a separate line
- Every **statement** ends with a semicolon

Magic recipes, Example 1

25

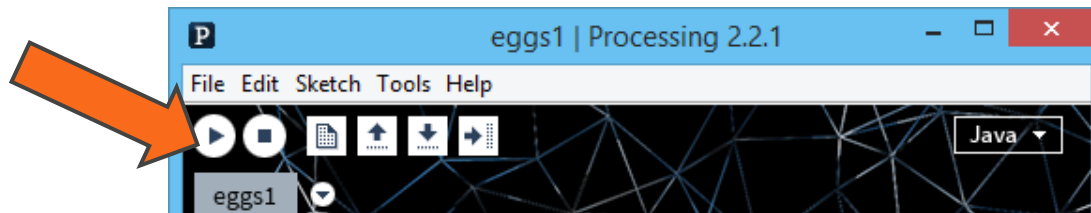
```
void draw() {  
    int bowl = 5;  
    drawEggs(bowl);  
}
```

- Let's try to understand this **block** as a recipe: This block says that the system should do **two things**
- **First:** let's get a container, name it "bowl" and put the number 5 inside
- **Second:** let's see what's inside the bowl container and use the number to draw some eggs on screen

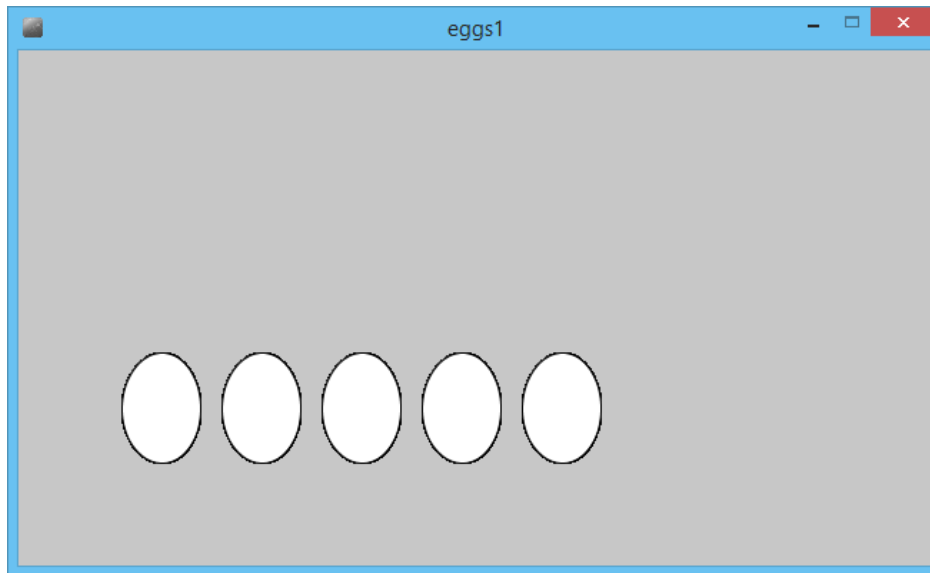
Magic recipes, Example 1

26

- Press “Play” to start the program



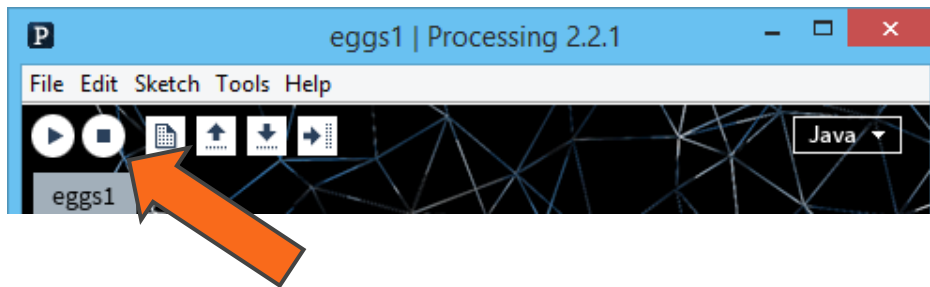
- Another window will appear (and some eggs! :)



Magic recipes, Example 1

27

- Press “Stop” to stop the program



Magic recipes, Example 1

28

```
void draw() {  
    int bowl = 5;  
    drawEggs(bowl);  
}
```

- Change the code to display 7 eggs!
(or any number of eggs :)

Magic recipes, Example 1

29

```
void draw() {  
    int bowl = 5;  
    drawEggs(bowl);  
}
```

- Let's understand what these two **statements** do
- First statement: **declares** that **bowl** is a **variable** that hold **integer values** – think of it as a little box that we can put a number in it
- First statement: **assigns** a **value** to **bowl**, i.e., number 5

Magic recipes, Example 1

30

```
void draw() {  
    int bowl = 5;  
    drawEggs(bowl);  
}
```

- Let's understand what these two **statements** do
- Second statement: **calls a function**, one that has the name `drawEggs` and is provided for us to draw things
- Second statement: the function will look into the value of the variable `bowl` in order to know how many eggs to draw, this called an **argument**

Magic recipes, Example 2

31

- Download file eggs2.zip from the following link:
 - <http://tinyurl.com/int-mult-2015-pde>
- Unzip the file and open eggs2.pde from folder eggs2



The screenshot shows the Processing IDE window titled 'eggs2 | Processing 2.2.1'. The code in the editor is as follows:

```
void setup() {  
  size(640, 360);  
  background(199);  
  fill(255);  
}  
  
void draw() {  
  int bowl = 2;  
  int anotherbowl = 4;  
  bowl = anotherbowl;  
  //bowl = anotherbowl + 1;  
  //bowl = bowl + 1  
  drawEggs(bowl);  
}  
  
void drawEggs(int n) {  
  for (int i = 0; i < n; i++) {  
    ellipse(100+i*70, 250, 55, 77);  
  }  
}
```

Magic recipes, Example 2

32

```
void draw() {  
    int bowl = 2;  
    int anotherbowl = 4;  
    bowl = anotherbowl;  
    //bowl = anotherbowl + 1;  
    //bowl = bowl + 1;  
    drawEggs (bowl) ;  
}
```


Magic recipes, Example 2

33

```
void draw() {  
    int bowl = 2;  
    int anotherbowl = 4;  
    bowl = anotherbowl;  
    //bowl = anotherbowl + 1;  
    //bowl = bowl + 1;  
    drawEggs (bowl) ;  
}
```

- ▣ Notice the lines with different color
- ▣ These are “comments” which the program will ignore
- ▣ It is an easy way to try out different things

Magic recipes, Example 2

34

```
void draw() {  
    int bowl = 2;  
    int anotherbowl = 4;  
    //bowl = anotherbowl;  
    bowl = anotherbowl + 1;  
    //bowl = bowl + 1;  
    drawEggs (bowl) ;  
}
```

- ▣ “Comment” and “Uncomment” the statements that assign values to the variable **bowl** to run different variants
- ▣ How many eggs will be drawn now?

Magic recipes, Example 2

35

```
void draw() {  
    int bowl = 2;  
    int anotherbowl = 4;  
    //bowl = anotherbowl;  
    //bowl = anotherbowl + 1;  
    bowl = bowl + 1;  
    drawEggs (bowl) ;  
}
```

- ▣ “Comment” and “Uncomment” the statements that assign values to the variable **bowl** to run different variants
- ▣ How many eggs will be drawn now?

Magic recipes, Example 2

36

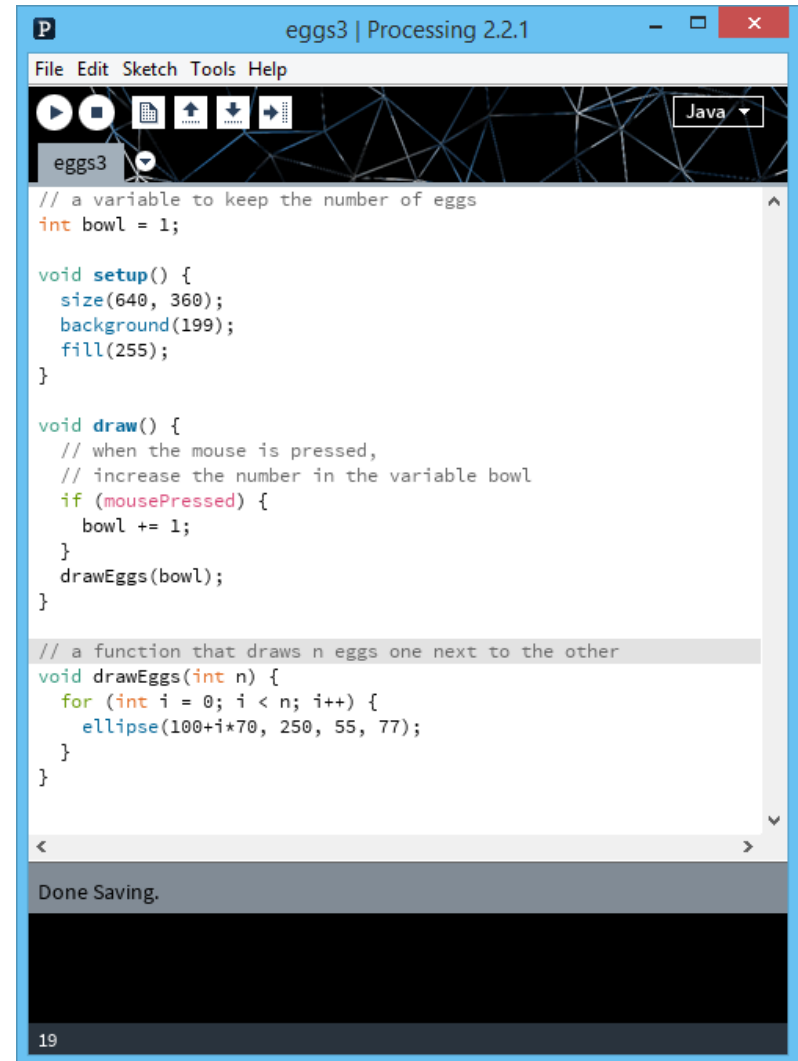
```
void draw() {  
    int bowl = 2;  
    int anotherbowl = 4;  
    bowl = anotherbowl;  
    bowl = anotherbowl + 1;  
    bowl = bowl + 1;  
    drawEggs (bowl) ;  
}
```

- ▣ “Comment” and “Uncomment” the statements that assign values to the variable **bowl** to run different variants
- ▣ How many eggs will be drawn now?

Magic recipes, Example 3

37

- Download file eggs3.zip from the following link:
 - <http://tinyurl.com/int-mult-2015-pde>
- Unzip the file and open eggs3.pde from folder eggs3



The screenshot shows the Processing 2.2.1 IDE window titled 'eggs3 | Processing 2.2.1'. The code in the editor is as follows:

```
File Edit Sketch Tools Help
eggs3
// a variable to keep the number of eggs
int bowl = 1;

void setup() {
  size(640, 360);
  background(199);
  fill(255);
}

void draw() {
  // when the mouse is pressed,
  // increase the number in the variable bowl
  if (mousePressed) {
    bowl += 1;
  }
  drawEggs(bowl);
}

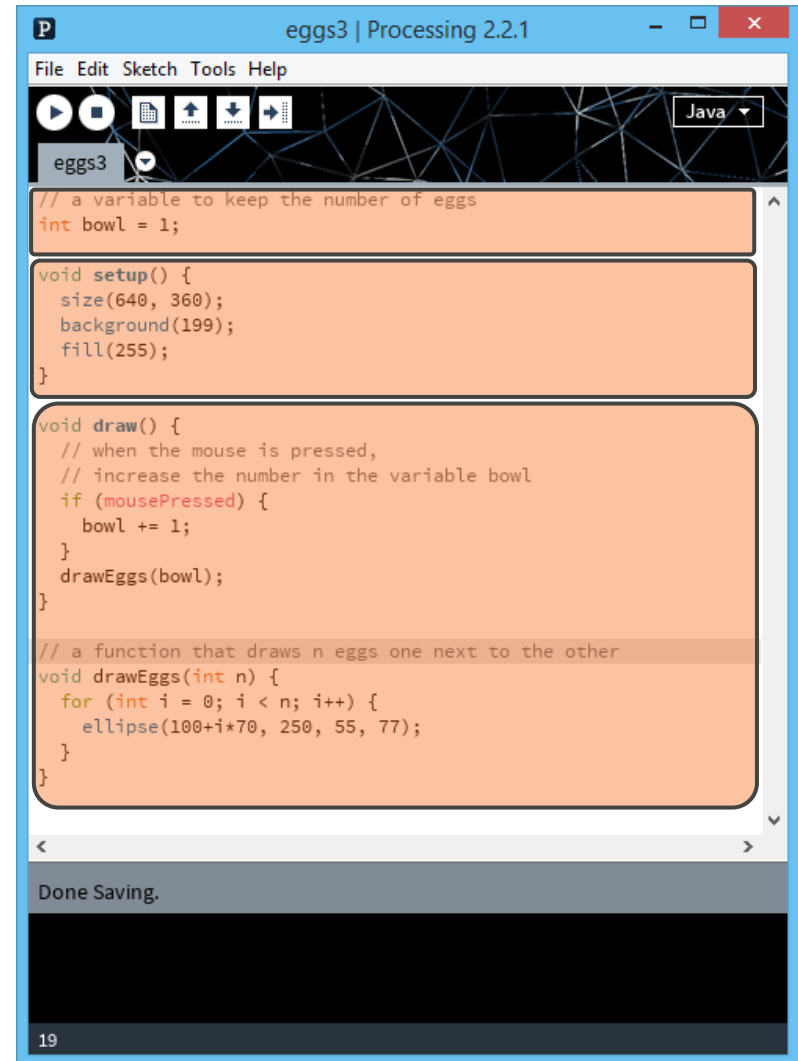
// a function that draws n eggs one next to the other
void drawEggs(int n) {
  for (int i = 0; i < n; i++) {
    ellipse(100+i*70, 250, 55, 77);
  }
}
```

At the bottom of the IDE, a status bar displays 'Done Saving.' and the page number '19'.

Magic recipes, Example 3

38

- There are three main parts
 - ▣ A part where we declare variables
 - ▣ A part where we initialize things once
 - ▣ And a part where the say what should happen inside a continuous loop



```
eggs3 | Processing 2.2.1
File Edit Sketch Tools Help
eggs3
// a variable to keep the number of eggs
int bowl = 1;

void setup() {
  size(640, 360);
  background(199);
  fill(255);
}

void draw() {
  // when the mouse is pressed,
  // increase the number in the variable bowl
  if (mousePressed) {
    bowl += 1;
  }
  drawEggs(bowl);
}

// a function that draws n eggs one next to the other
void drawEggs(int n) {
  for (int i = 0; i < n; i++) {
    ellipse(100+i*70, 250, 55, 77);
  }
}

Done Saving.
19
```

Magic recipes, Example 3

39

- There are three main parts
 - ▣ A part where we **declare variables**
 - ▣ A part where we initialize things once
 - ▣ And a part where the say what should happen inside a continuous loop

A screenshot of the Processing 2.2.1 IDE. The window title is "eggs3 | Processing 2.2.1". The menu bar includes "File", "Edit", "Sketch", "Tools", and "Help". The toolbar contains icons for play, stop, refresh, and other sketch functions. The code editor shows the following code:

```
// a variable to keep the number of eggs
int bowl = 1;

void setup() {
  size(640, 360);
  background(199);
  fill(255);
}

void draw() {
  // when the mouse is pressed,
  // increase the number in the variable bowl
  if (mousePressed) {
    bowl += 1;
  }
  drawEggs(bowl);
}

// a function that draws n eggs one next to the other
void drawEggs(int n) {
  for (int i = 0; i < n; i++) {
    ellipse(100+i*70, 250, 55, 77);
  }
}
```

The first line of code, `// a variable to keep the number of eggs`, and the second line, `int bowl = 1;`, are highlighted in orange. The IDE also shows a "Done Saving." message at the bottom and the number "19" in the bottom left corner.

Magic recipes, Example 3

40

```
// a variable to keep the number of eggs  
int bowl;
```


Magic recipes, Example 3

41

- There are three main parts
 - ▣ A part where we declare variables
 - ▣ A part where we **initialize things once**
 - ▣ And a part where the say what should happen inside a continuous loop



```
eggs3 | Processing 2.2.1
File Edit Sketch Tools Help
eggs3
// a variable to keep the number of eggs
int bowl = 1;

void setup() {
  size(640, 360);
  background(199);
  fill(255);
}

void draw() {
  // when the mouse is pressed,
  // increase the number in the variable bowl
  if (mousePressed) {
    bowl += 1;
  }
  drawEggs(bowl);
}

// a function that draws n eggs one next to the other
void drawEggs(int n) {
  for (int i = 0; i < n; i++) {
    ellipse(100+i*70, 250, 55, 77);
  }
}

Done Saving.
19
```

Magic recipes, Example 3

42

```
// initialize the size of the window
size(640, 360);
// initialize the background
background(199);
// initialize the color for shapes
fill(255);
// initialize the variable bowl
bowl = 1;
```

Magic recipes, Example 3

43

```
// a variable to keep the number of eggs  
int bowl;
```

```
void setup() {  
  size(640, 360);  
  background(199);  
  fill(255);  
  bowl = 1;  
}
```

Magic recipes, Example 3

44

- There are three main parts
 - ▣ A part where we declare variables
 - ▣ A part where we initialize things once
 - ▣ And a part where the say what should happen inside a **continuous loop**



```
Processing 2.2.1 | eggs3 | Processing 2.2.1
File Edit Sketch Tools Help
eggs3
// a variable to keep the number of eggs
int bowl = 1;

void setup() {
  size(640, 360);
  background(199);
  fill(255);
}

void draw() {
  // when the mouse is pressed,
  // increase the number in the variable bowl
  if (mousePressed) {
    bowl += 1;
  }
  drawEggs(bowl);
}

// a function that draws n eggs one next to the other
void drawEggs(int n) {
  for (int i = 0; i < n; i++) {
    ellipse(100+i*70, 250, 55, 77);
  }
}
```

Done Saving.

19

Magic recipes, Example 3

45

```
void draw() {
    // when the mouse is pressed, increase
    // the number in the variable bowl
    if (mousePressed) {
        bowl += 1;
    }
    // use a given function to draw as
    // many eggs as in the variable bowl
    drawEggs (bowl) ;
}
```

Magic recipes, Example 3


46

```
void draw() {  
    if (mousePressed) {  
        bowl += 1;  
    }  
    drawEggs (bowl) ;  
}
```

Magic recipes, Example 3

47

```
void draw() {  
    if (mousePressed) {  
        bowl += 1;  
    }  
    drawEggs (bowl) ;  
}
```

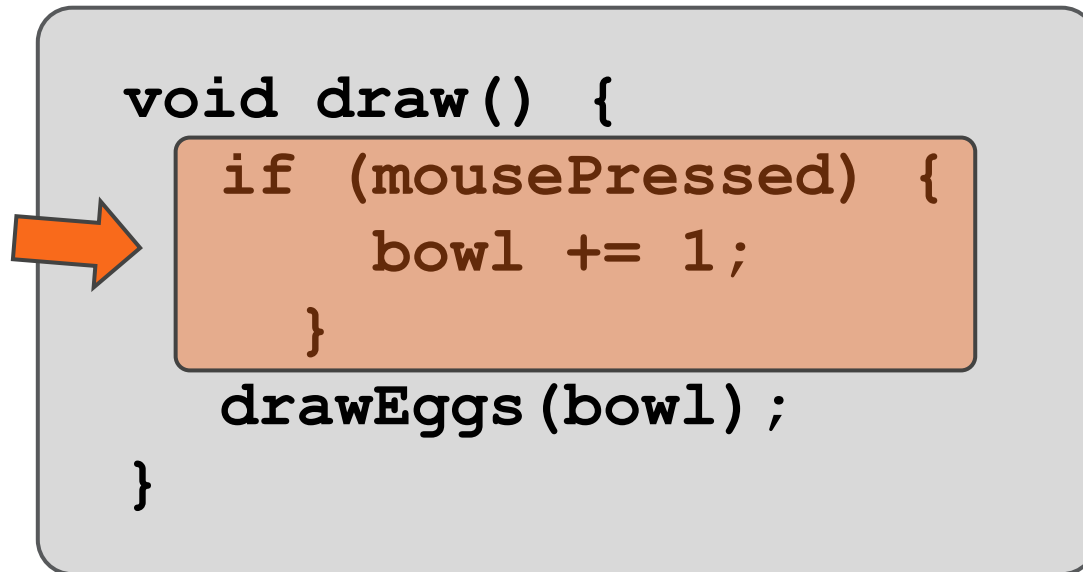


- ▣ There are a lot of “short hand” abbreviations in computer programming to help us write concise programs
- ▣ This is exactly the same as writing: `bowl = bowl + 1;`

Magic recipes, Example 3

48

```
void draw() {  
    if (mousePressed) {  
        bowl += 1;  
    }  
    drawEggs (bowl) ;  
}
```

The diagram illustrates a nested code block. An outer gray rounded rectangle contains the code for a `draw()` function. Inside this function, there is an `if (mousePressed)` block. This inner block is highlighted with an orange background and is pointed to by an orange arrow from the left. The code inside the `if` block is `bowl += 1;`. Below the `if` block, the code `drawEggs (bowl) ;` is shown, followed by the closing brace of the `draw()` function.

- ▣ A block inside a block!
- ▣ We have to get used to these! :)

Magic recipes, Example 3

49

```
void draw() {  
  if (mousePressed) {  
    bowl += 1;  
  }  
  drawEggs (bowl) ;  
}
```

- ❑ Here we use a conditional effect using the keyword **if**
- ❑ If the condition in parenthesis is true then the block of statements is executed
- ❑ <https://processing.org/reference/if.html>

Magic recipes, Example 3

50

```
void draw() {  
    if (mousePressed) {  
        bowl += 1;  
    }  
    drawEggs (bowl) ;  
}
```

- ▣ And what about this function we used to draw eggs?
- ▣ This was in fact a function we wrote, sort of a small recipe that we can use whenever we want

Magic recipes, Example 3

51

```
// a function that draws n eggs
// one next to the other
void drawEggs(int n) {
    for (int i = 0; i < n; i++) {
        ellipse(100+i*70, 250, 55, 77);
    }
}
```

- ▣ It's just a simple block that explains what should happen when we call the function **drawEggs**

Magic recipes, Example 3

52

```
// a function that draws n eggs
// one next to the other
void drawEggs(int n) {
  for (int i = 0; i < n; i++) {
    ellipse(100+i*70, 250, 55, 77);
  }
}
```

- ❑ Here we use an iterative loop with the keyword **for**
- ❑ This executes the block of statements as many times as specified in the parenthesis
- ❑ <https://processing.org/reference/for.html>

Magic recipes, Example 3

53

```
// a function that draws n eggs
// one next to the other
void drawEggs(int n) {
  for (int i = 0; i < n; i++) {
    ellipse(100+i*70, 250, 55, 77);
  }
}
```

- ▣ We also draw a simple ellipse, i.e., the egg, with the function `ellipse()`
- ▣ https://processing.org/reference/ellipse_.html

Magic recipes, Processing

54

- How do we know which are the available keywords and functions?

Magic recipes, Processing

55

- The whole “spell book” of Processing
 - ▣ <https://processing.org/reference/>



Cover

Download

Exhibition

Reference

Libraries

Tools

Environment

Tutorials

Reference. The Processing Language was designed to facilitate the creation of sophisticated visual structures.

Structure

() (parentheses)
, (comma)
. (dot)
/* */ (multiline comment)
/** */ (doc comment)
// (comment)

Shape

createShape()
loadShape()
PShape

2D Primitives
arc()

Color

Setting
background()
clear()
colorMode()
fill()
noFill()

Magic recipes, Processing

56

- Extra “spell books” that can be used, called **libraries**
 - ▣ <https://processing.org/reference/libraries/>

Processing p5.js Processing.py Processing Foundation

Processing

Cover

Download

Exhibition

Reference Libraries

Tools

Environment

Tutorials

Libraries. Extend Processing beyond graphics and images into audio, video, and communication with other devices.

The following libraries are created by the Processing Foundation. The PDF Export, Network, Serial, and DXF Export libraries are distributed with Processing. The Video and Sound libraries need to be downloaded through the Library Manager. Select "Add Library..." from the "Import Library..." submenu within the Sketch menu.

| | | |
|---|--|---|
| PDF Export Create PDF files. These vector graphics files can be scaled to any size and printed at high resolutions. | Serial Send data between Processing and external hardware through serial communication (RS-232). | Video Read images from a camera, play movie files, and create movies. |
|---|--|---|

Magic recipes, Processing

57

- Extra “spell books” that can be used, called **libraries**
 - <https://processing.org/reference/libraries/>
 - Go over the list and see all the possibilities for interaction!

Processing p5.js Processing.py Processing Foundation

Processing

Search

[Cover](#)

[Download](#)

[Exhibition](#)

[Reference Libraries](#)

[Tools](#)

[Environment](#)

[Tutorials](#)

Libraries. Extend Processing beyond graphics and images into audio, video, and communication with other devices.

The following libraries are created by the Processing Foundation. The PDF Export, Network, Serial, and DXF Export libraries are distributed with Processing. The Video and Sound libraries need to be downloaded through the Library Manager. Select "Add Library..." from the "Import Library..." submenu within the Sketch menu.

| | | |
|---|--|---|
| PDF Export Create PDF files. These vector graphics files can be scaled to any size and printed at high resolutions. | Serial Send data between Processing and external hardware through serial communication (RS-232). | Video Read images from a camera, play movie files, and create movies. |
|---|--|---|

Magic recipes, Processing

58

- Small (relatively simple) examples for many scenarios
 - <https://processing.org/examples/>

Cover

Download

Exhibition

Reference

Libraries

Tools

Environment

Examples. Short, prototypical programs exploring the basics of programming with Processing.

These examples are running online through Processing.js using HTML5 Canvas and WebGL for rendering. There are many more examples included with the Processing application; please look there if you don't find what you're looking for here.

Basic Examples. Programs about form, data, images, color, typography, and more...

Tutorials

Examples

Books

Handbook

Overview

People

Shop

Structure

Statements and Comments

Coordinates

Width and Height

Setup and Draw

No Loop

Loop

Redraw

Image

Load and Display Image

Background Image

Transparency

Alphamask

CreateImage

Pointillism

Input

Mouse 1D

Mouse 2D

MousePress

Mouse Signals

Easing

Constrain

Storing Input



Magic recipes, Processing

59

- Tutorials for many scenarios
 - <https://processing.org/tutorials/>
 - Try them out! They are very helpful and go step by step



Cover

Download

Exhibition

Reference

Libraries

Tools

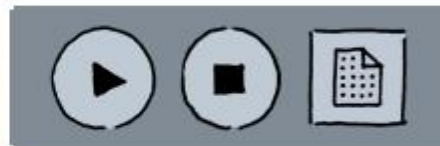
Environment

Tutorials

Tutorials. A collection of step-by-step lessons covering beginner, intermediate, and advanced topics.



[Hello Processing](#)
by Daniel Shiffman et al.



[Getting Started](#)
by Casey Reas and Ben Fry



[Processing Overview](#)
by Ben Fry and Casey Reas

Magic recipes, Processing

60

- For example here are some resources for using images in Processing
 - Tutorial: <https://processing.org/tutorials/pixels/>
 - Example “Load and Display Image”:
<https://processing.org/examples/loaddisplayimage.html>
 - Example “Background Image”:
<https://processing.org/examples/backgroundimage.html>
 - Example “Pointillism”:
<https://processing.org/examples/pointillism.html>
 - Reference for function `image()`:
https://processing.org/reference/image_.html