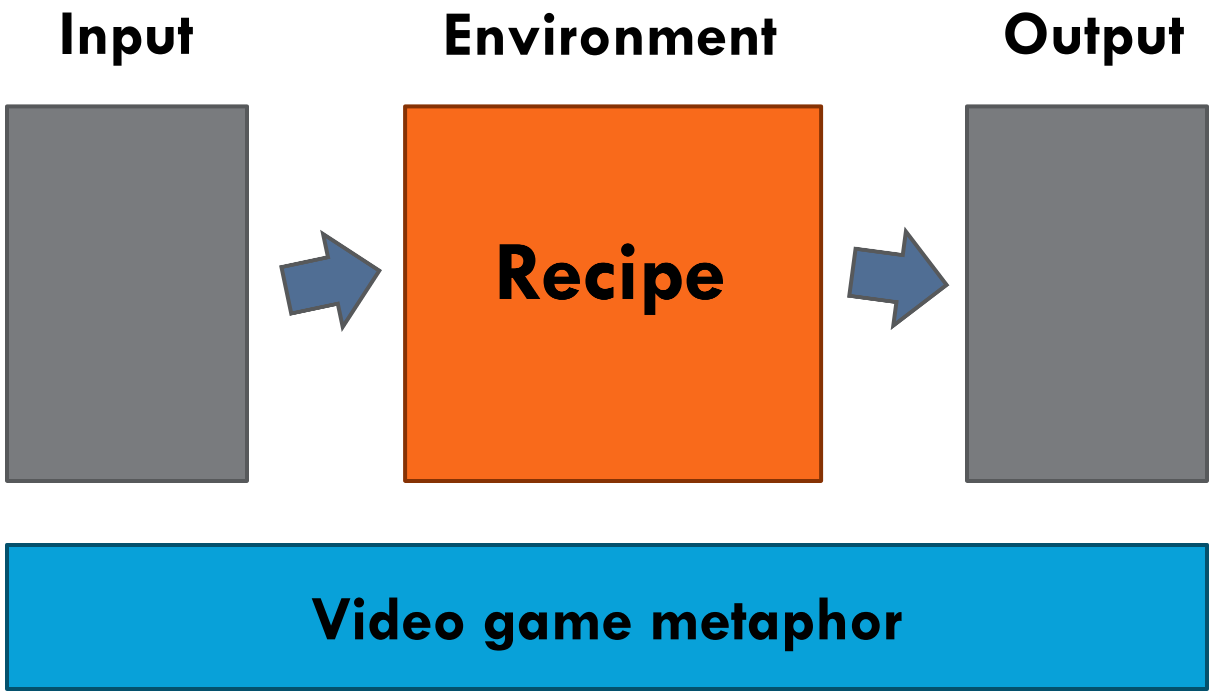
Stavros Vassos Sapienza University of Rome, DIAG, Italy vassos@dis.uniroma1.it Spring 2016

INTERACTIVE MULTIMEDIA DESIGN

With an AI touch :)

Interactive Multimedia Design

- It's important to separate the inputs and outputs



Computer programming

Computer programming

Computer programming

Computer programming

Computer programming

- 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

- 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

- *"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

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

Computer programming

- 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

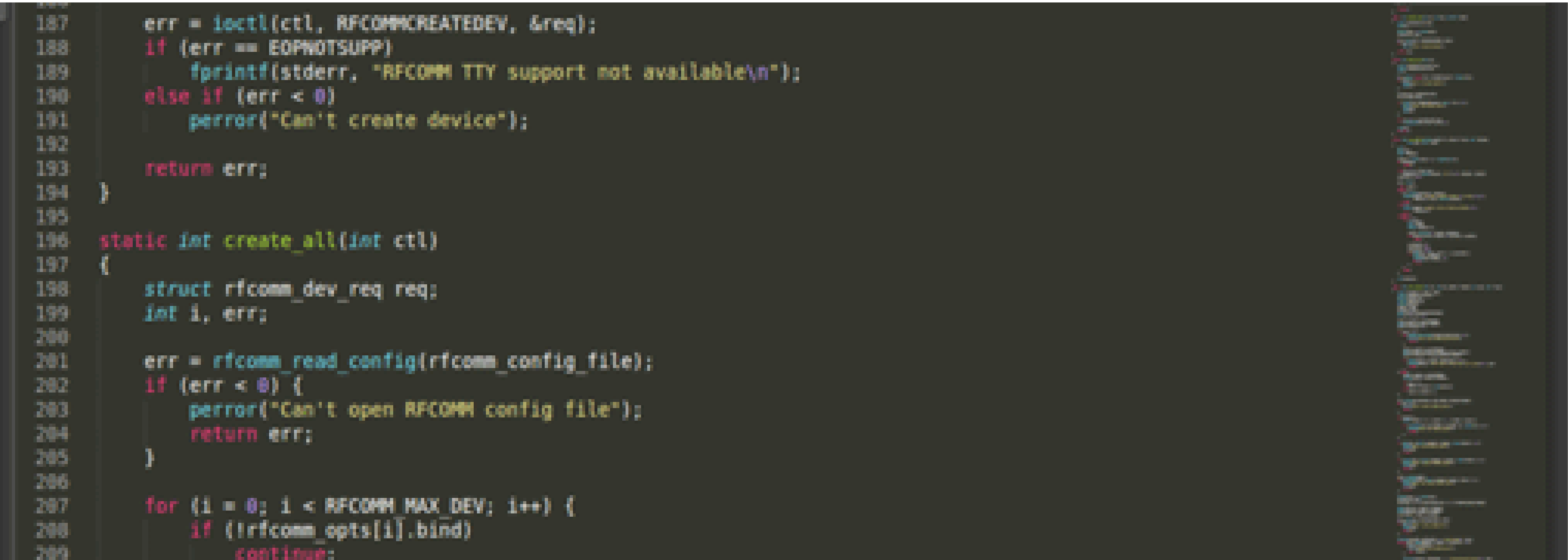
- We will look over some basic things to get you started with programming
- There is an **enormous** amount of information **online**, there is always someone who had a similar challenge, and usually there is documentation for it
- It is assumed that you have no previous knowledge or experience of programming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

Computer programming

- 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 :-)

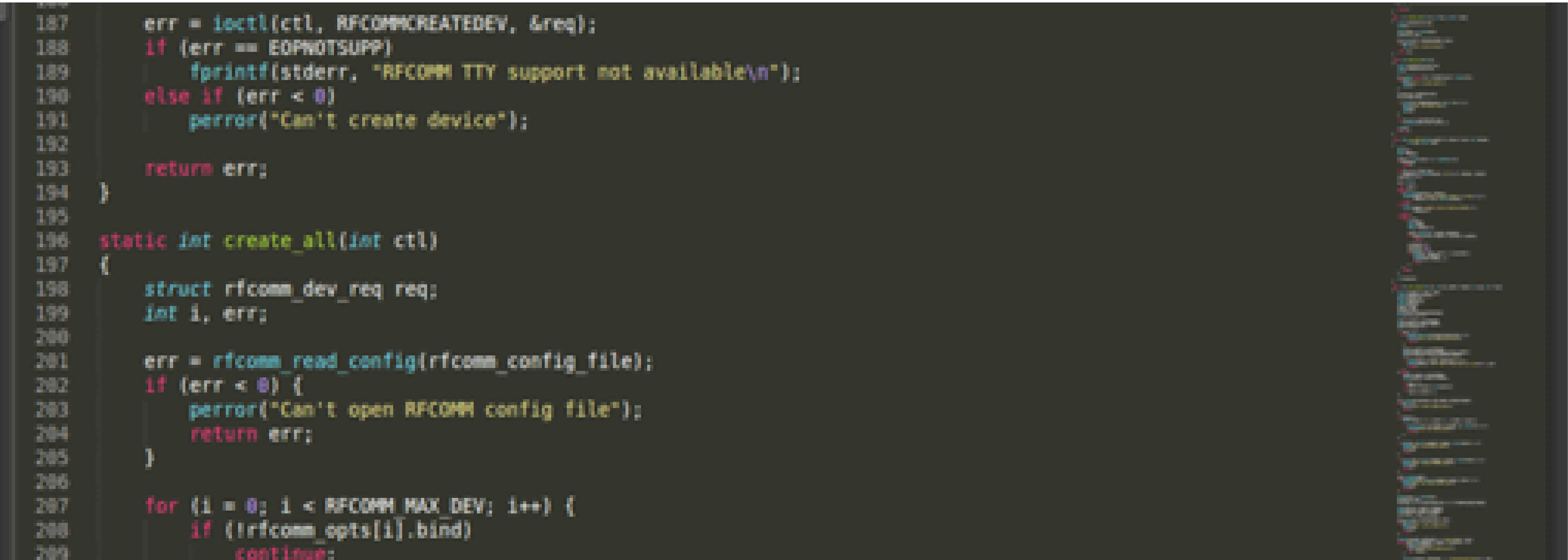
Source code

- We start by a **program** as a list of **statements** using the vocabulary of a programming language written as **simple text**, called **source code**



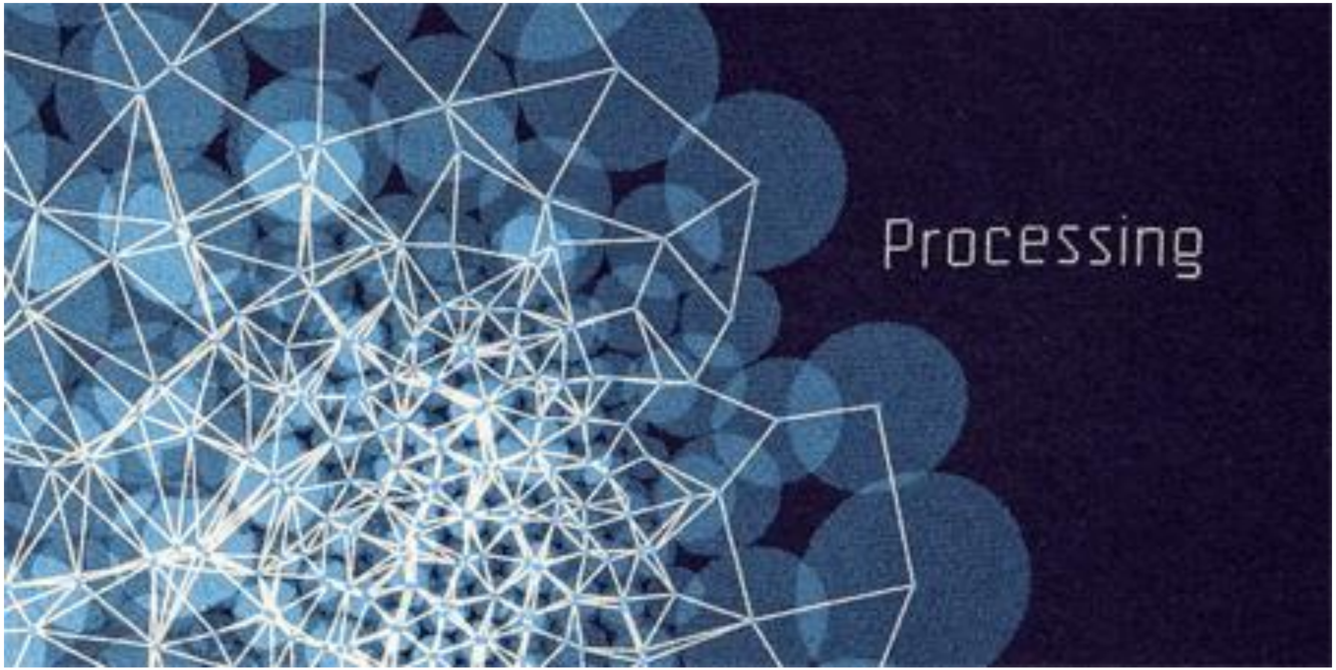
Source code

- Source code is used to produce an **output** which may be a **program** like the ones you use occasionally



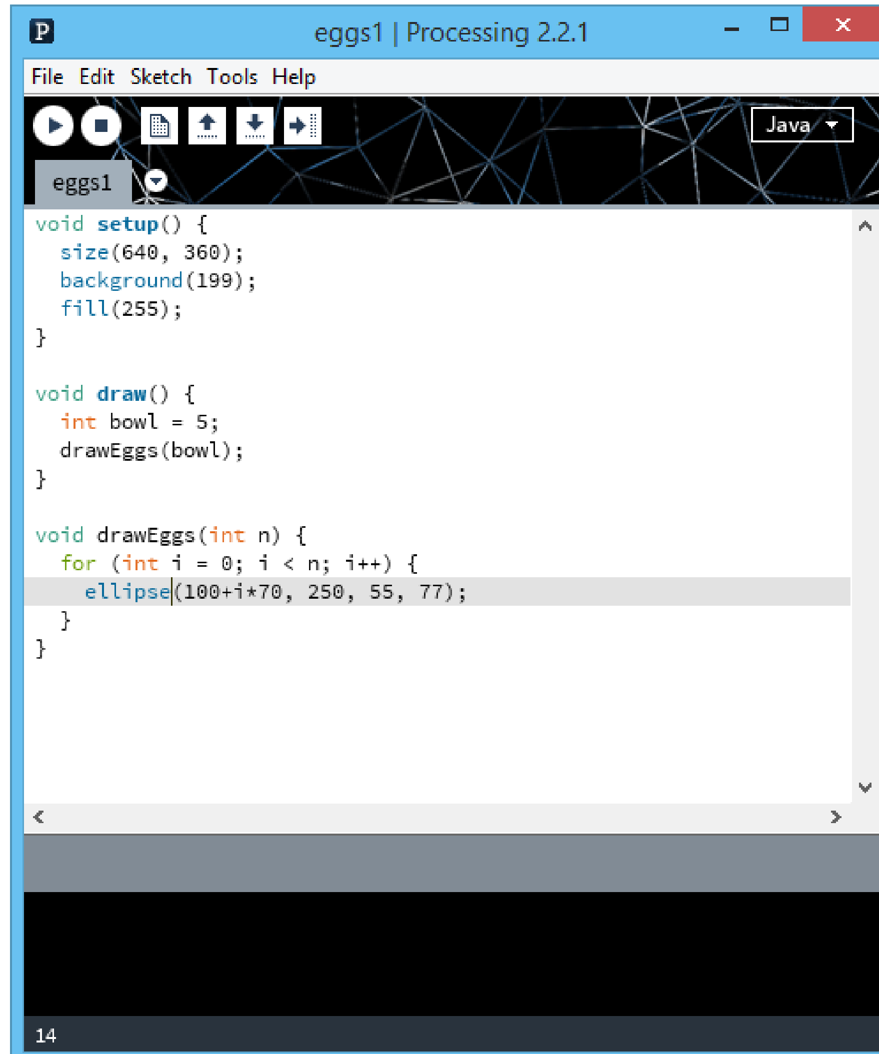
Processing programming language

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



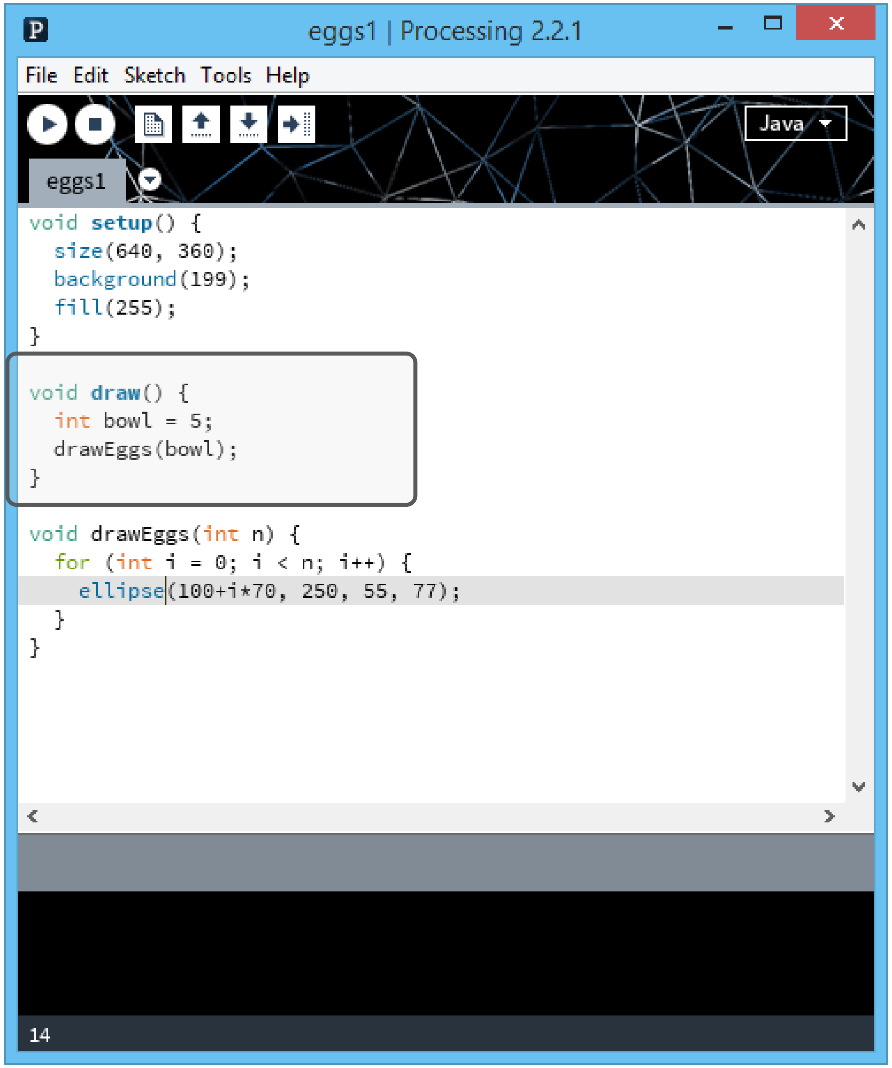
Magic recipes, Example 1

- 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



Magic recipes, Example 1

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



Magic recipes, Example 1


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

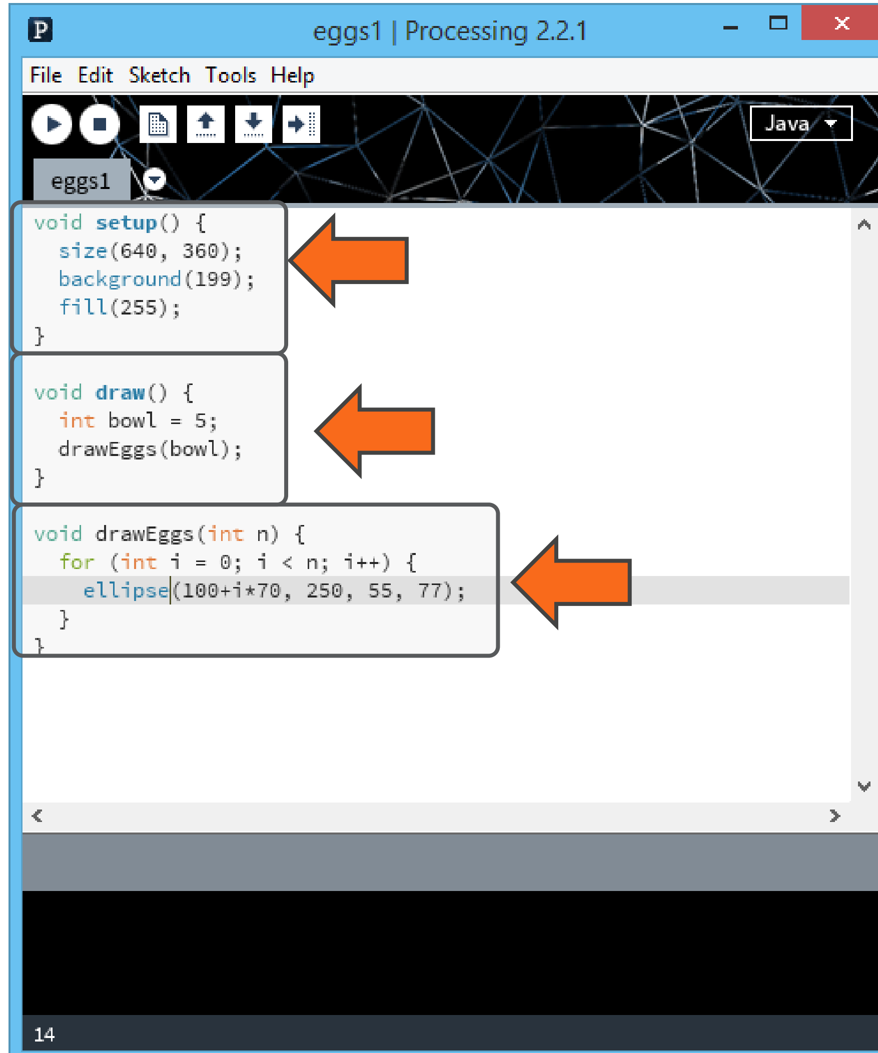
Magic recipes, Example 1


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

- 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

- 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



Magic recipes, Example 1


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

- 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


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

- This **block** has a name: "`void draw()`"

Magic recipes, Example 1


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

- 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

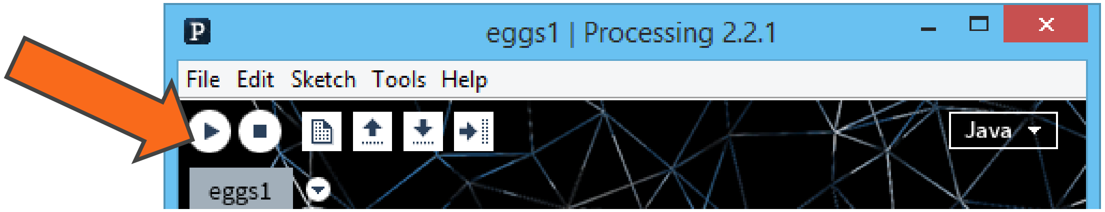
Magic recipes, Example 1


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

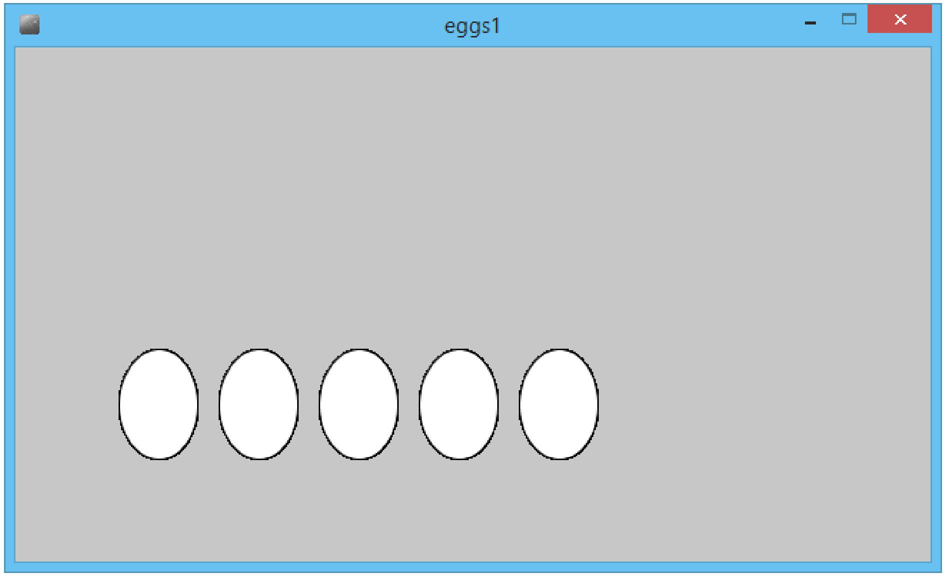
- Let’s try to understand this **block** as a recipe: This block says that the system should do **two things**
- **First:** let’s get a container, name it “bowl” and put the number 5 inside
- **Second:** let’s see what’s inside the bowl container and use the number to draw some eggs on screen

Magic recipes, Example 1

- Press “Play” to start the program

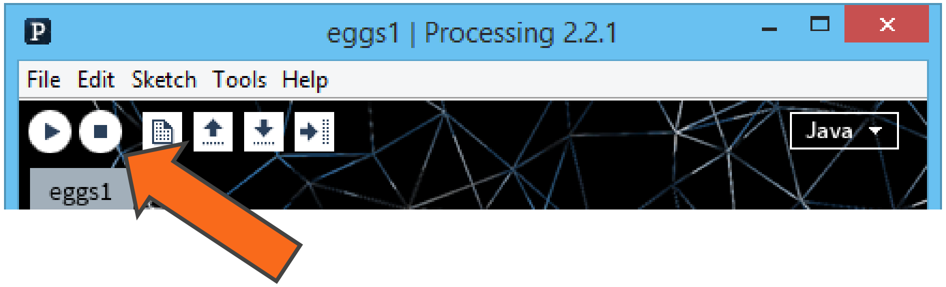


- Another window will appear (and some eggs! :)



Magic recipes, Example 1

- Press "Stop" to stop the program



Magic recipes, Example 1


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

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

Magic recipes, Example 1


```
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

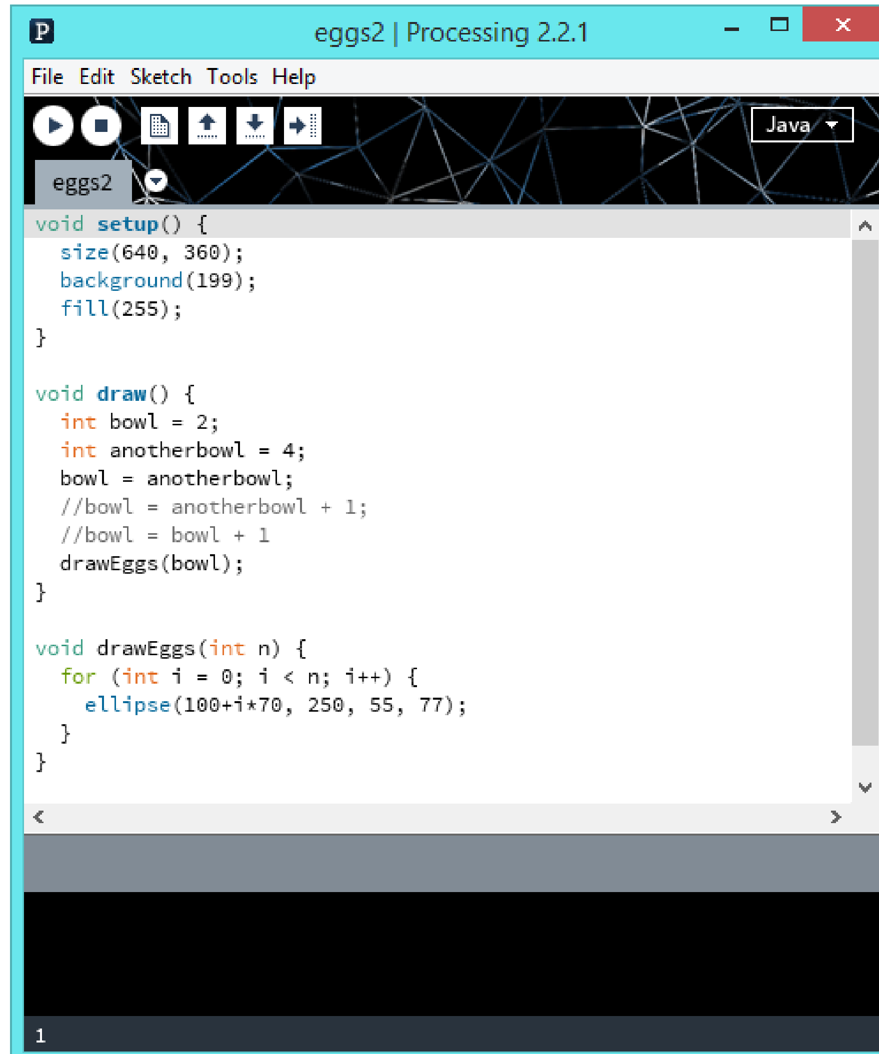

```
void draw() {
   int bowl = 5;
   drawEggs(bowl);
}
```

- 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

- 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



Magic recipes, Example 2


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

Magic recipes, Example 2


```
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


```
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


```
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

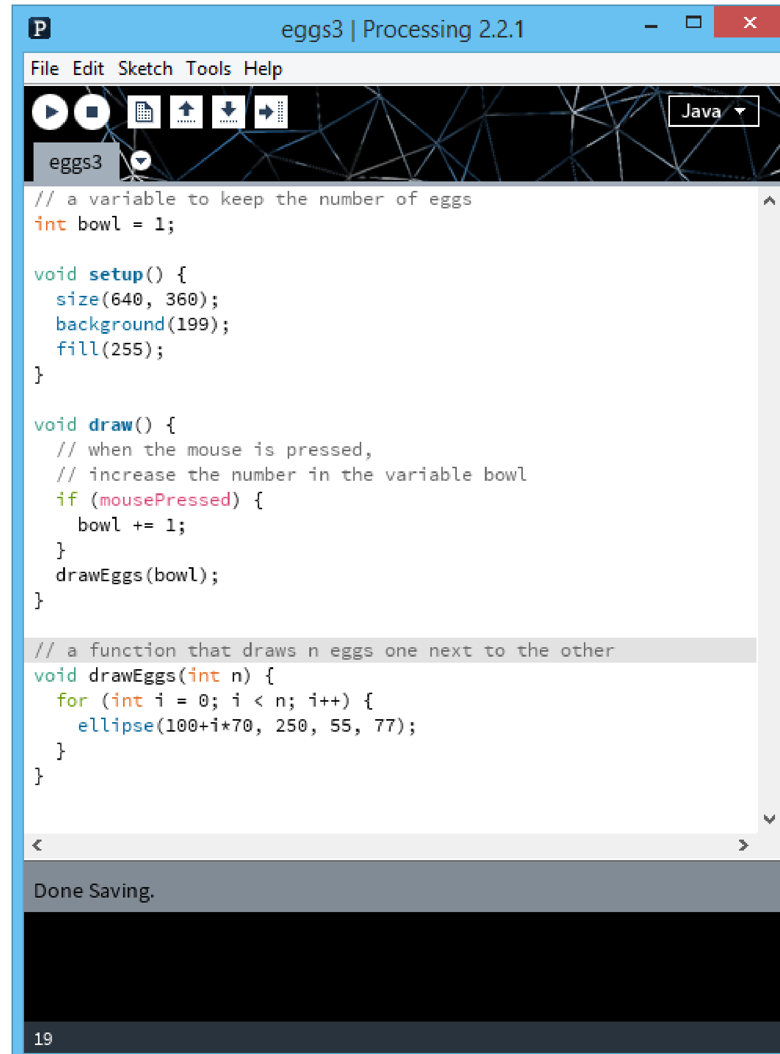

```
void draw() {
  int bowl = 2;
  int anotherbowl = 4;
  bowl = anotherbowl;
  bowl = anotherbowl + 1;
  bowl = bowl + 1;
  drawEggs(bowl);
}
```

- “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

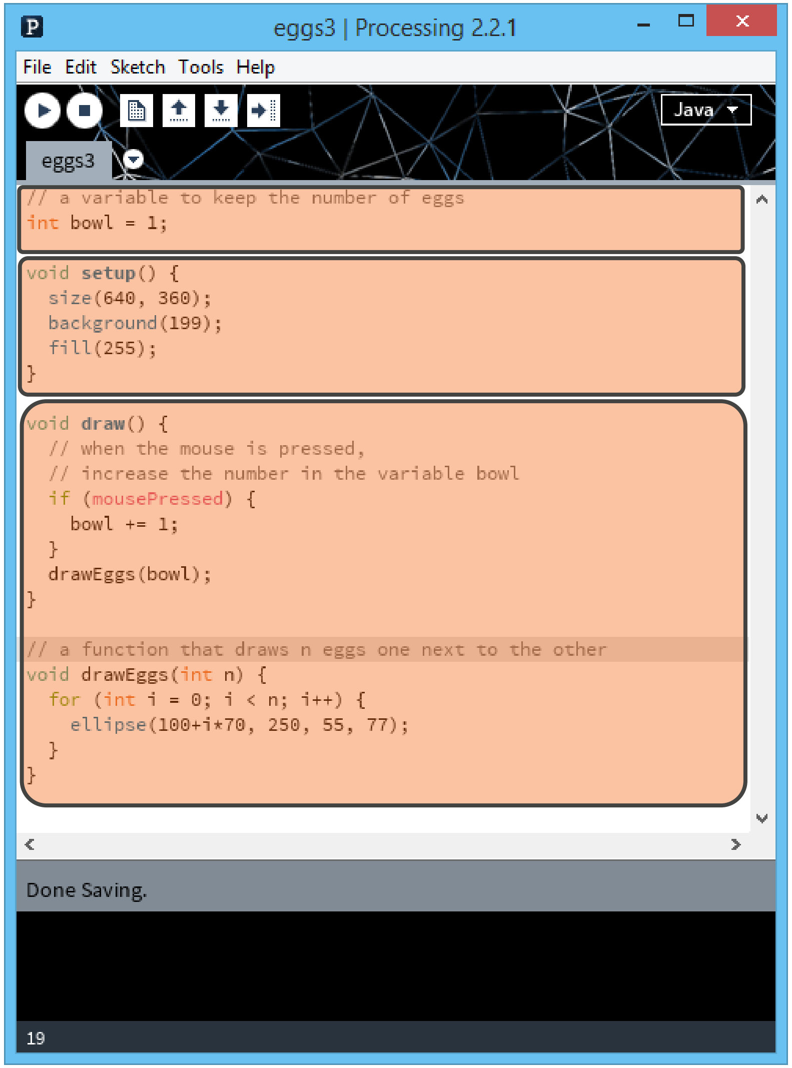
- Download file eggs3.zip from the following link:
  - http://tinyurl.com/int-mult-2015-pde

- Unzip the file and open eggs3.pde from folder eggs3



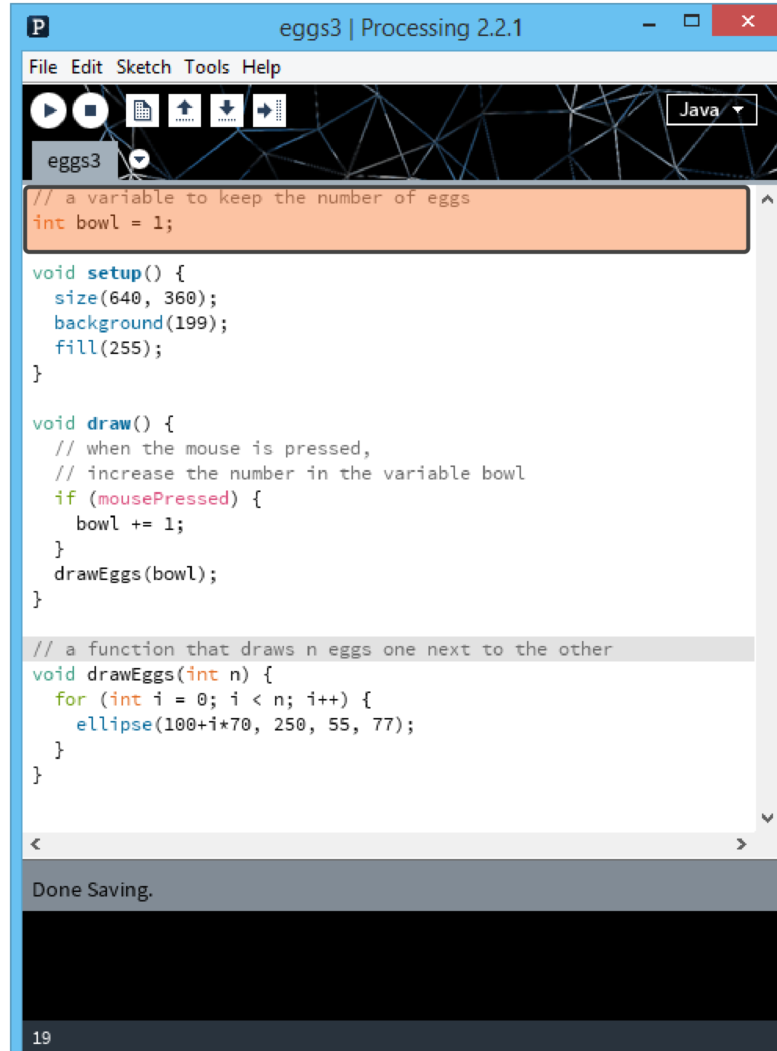
Magic recipes, Example 3

- 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



Magic recipes, Example 3

- 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

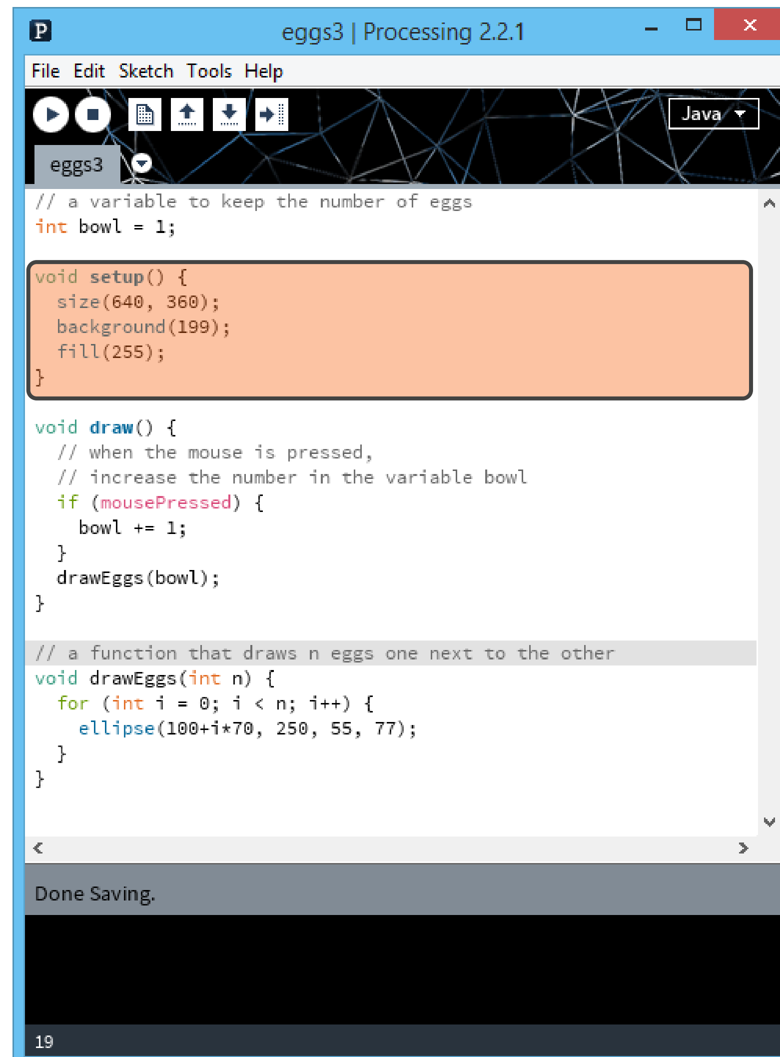


Magic recipes, Example 3


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

Magic recipes, Example 3

- 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



Magic recipes, Example 3


```
// 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;
```

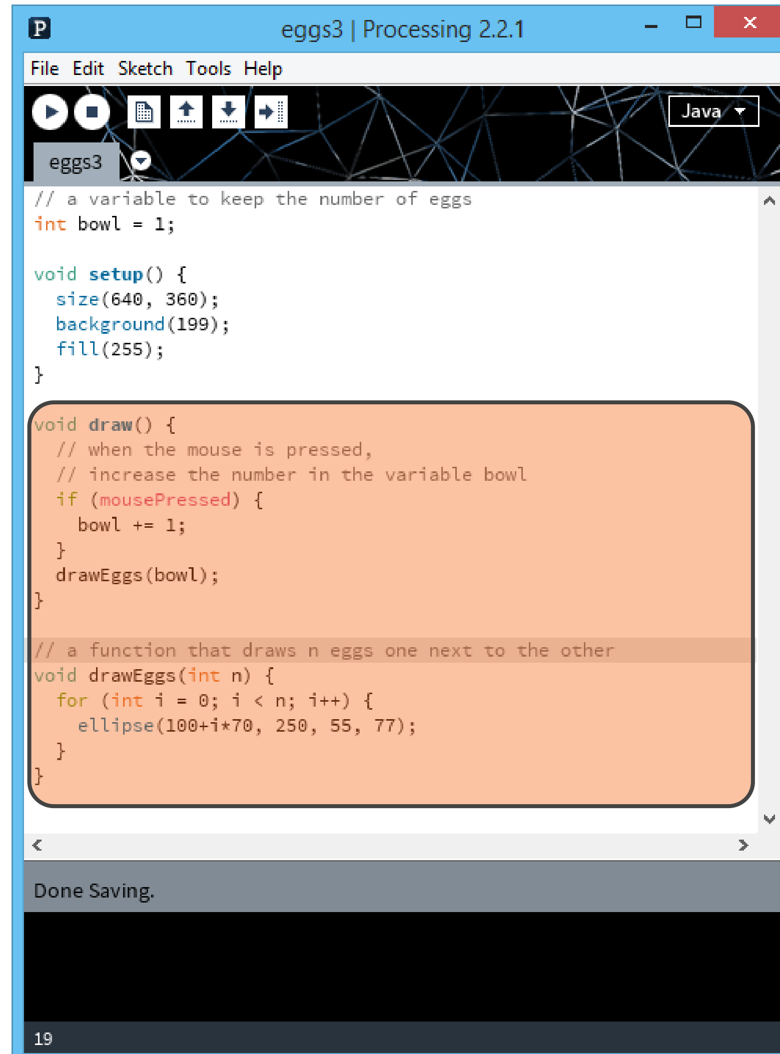
Magic recipes, Example 3


```
// 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

- 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**



Magic recipes, Example 3


```
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


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

Magic recipes, Example 3


```
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


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

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

Magic recipes, Example 3


```
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


```
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


```
// 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


```
// 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


```
// 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

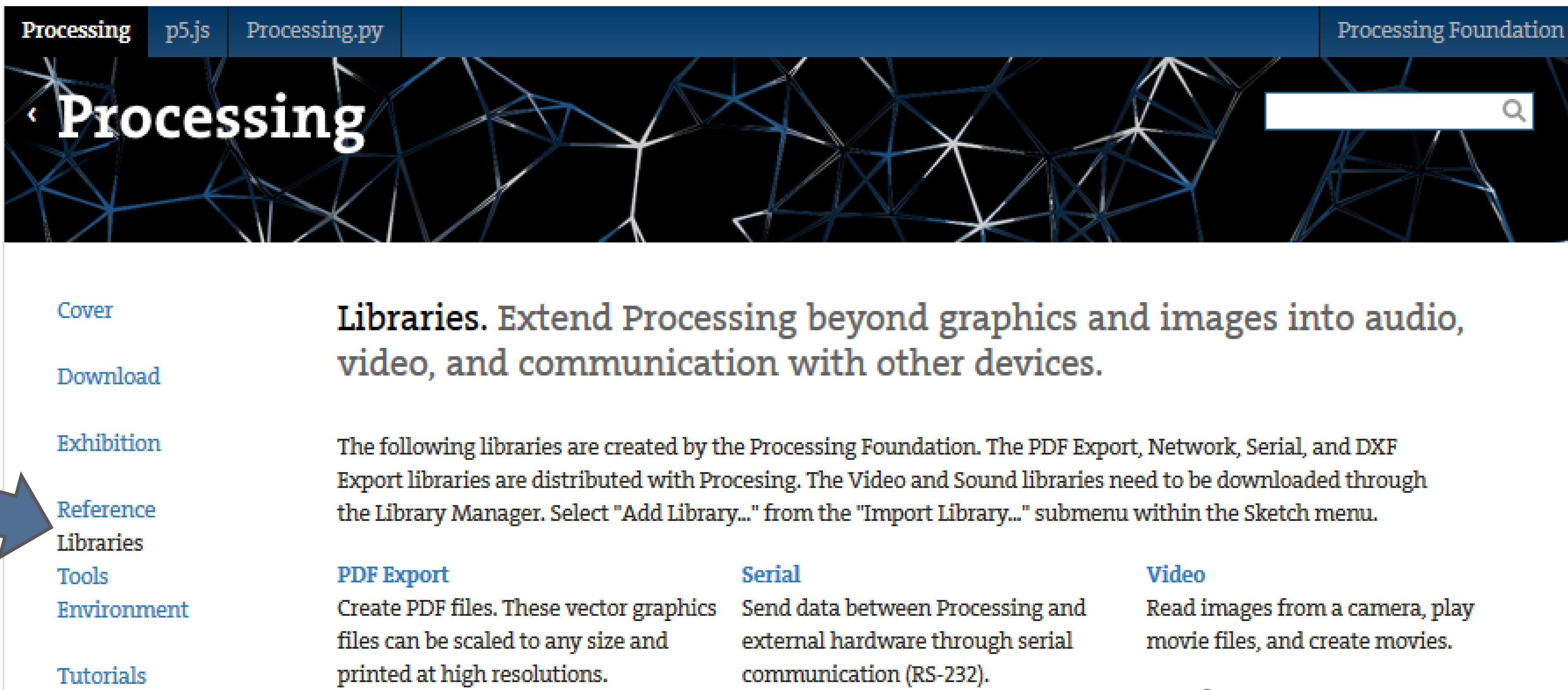
- How do we know which are the available keywords and functions?

Magic recipes, Processing

- The whole "spell book" of Processing
  - https://processing.org/reference/

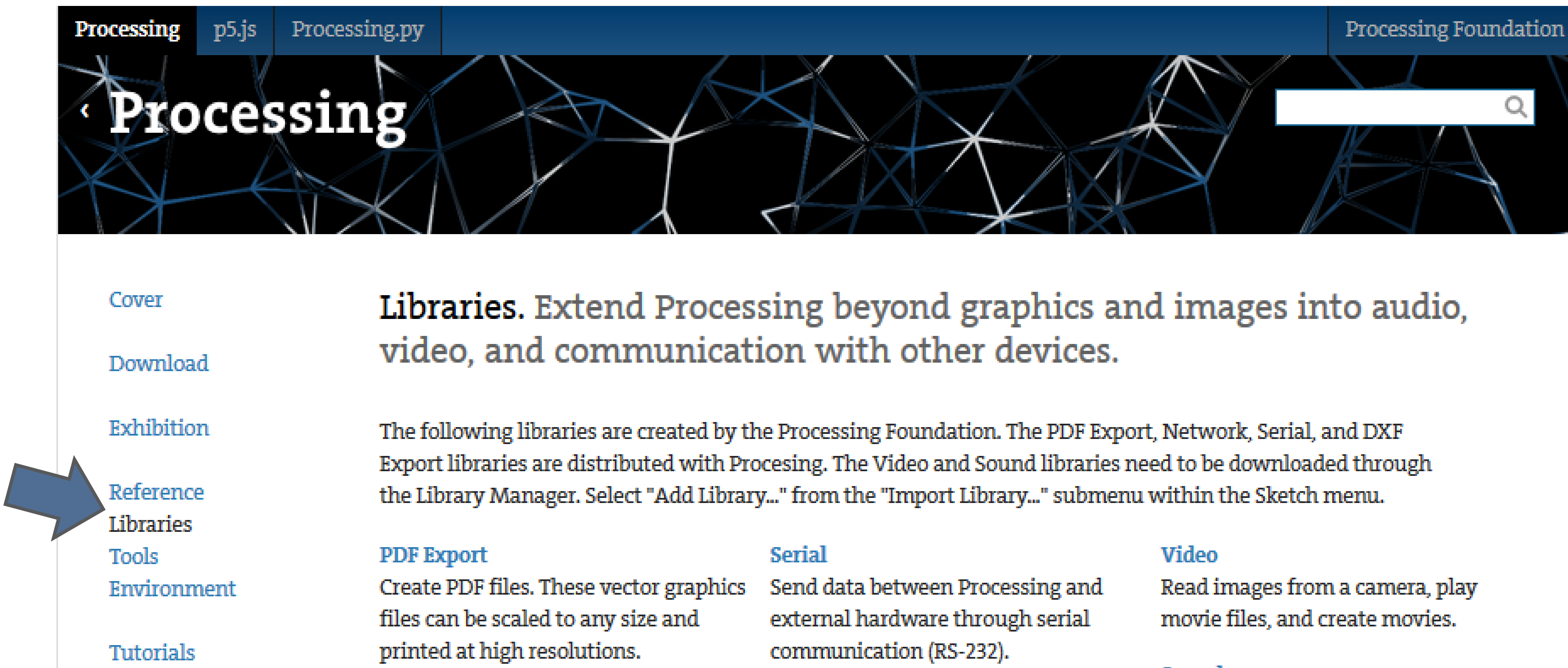
Magic recipes, Processing

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



Magic recipes, Processing

- 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!



Magic recipes, Processing

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

Cover

Download

Exhibition

Reference
Libraries
Tools
Environment

Tutorials
Examples
Books
Handbook

Overview
People

Shop

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

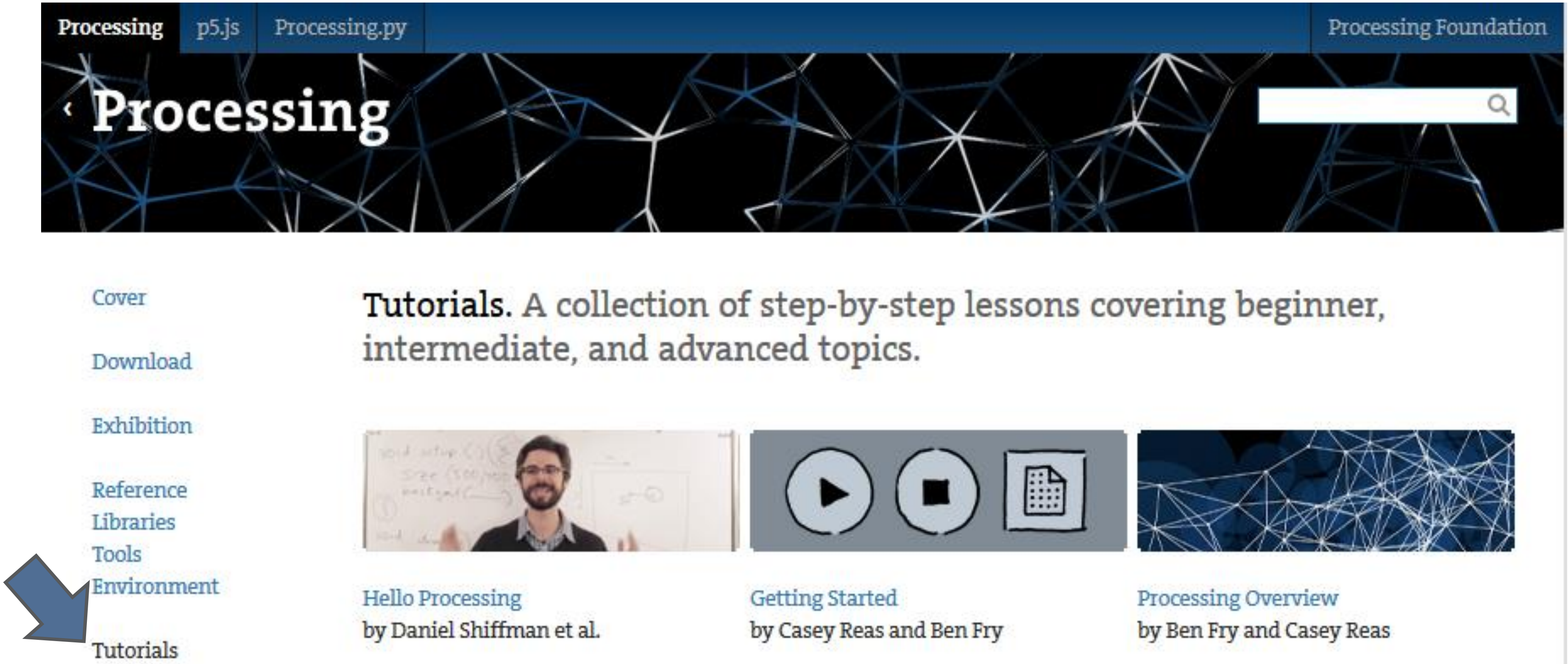
These examples are running online through Processing.js using HTML5 Canvas and WebGL for rendering. There are many more examples included with the Processing application; please look there if you don't find what you're looking for here.

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

Structure	Image	Input
Statements and Comments	Load and Display Image	Mouse 1D
Coordinates	Background Image	Mouse 2D
Width and Height	Transparency	MousePress
Setup and Draw	Alphamask	Mouse Signals
No Loop	CreateImage	Easing
Loop	Pointillism	Constrain
Redraw		Storing Input

Magic recipes, Processing

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



Magic recipes, Processing

- 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