POLITECNICO
MILANO 1863

Digital Art

2022-2023

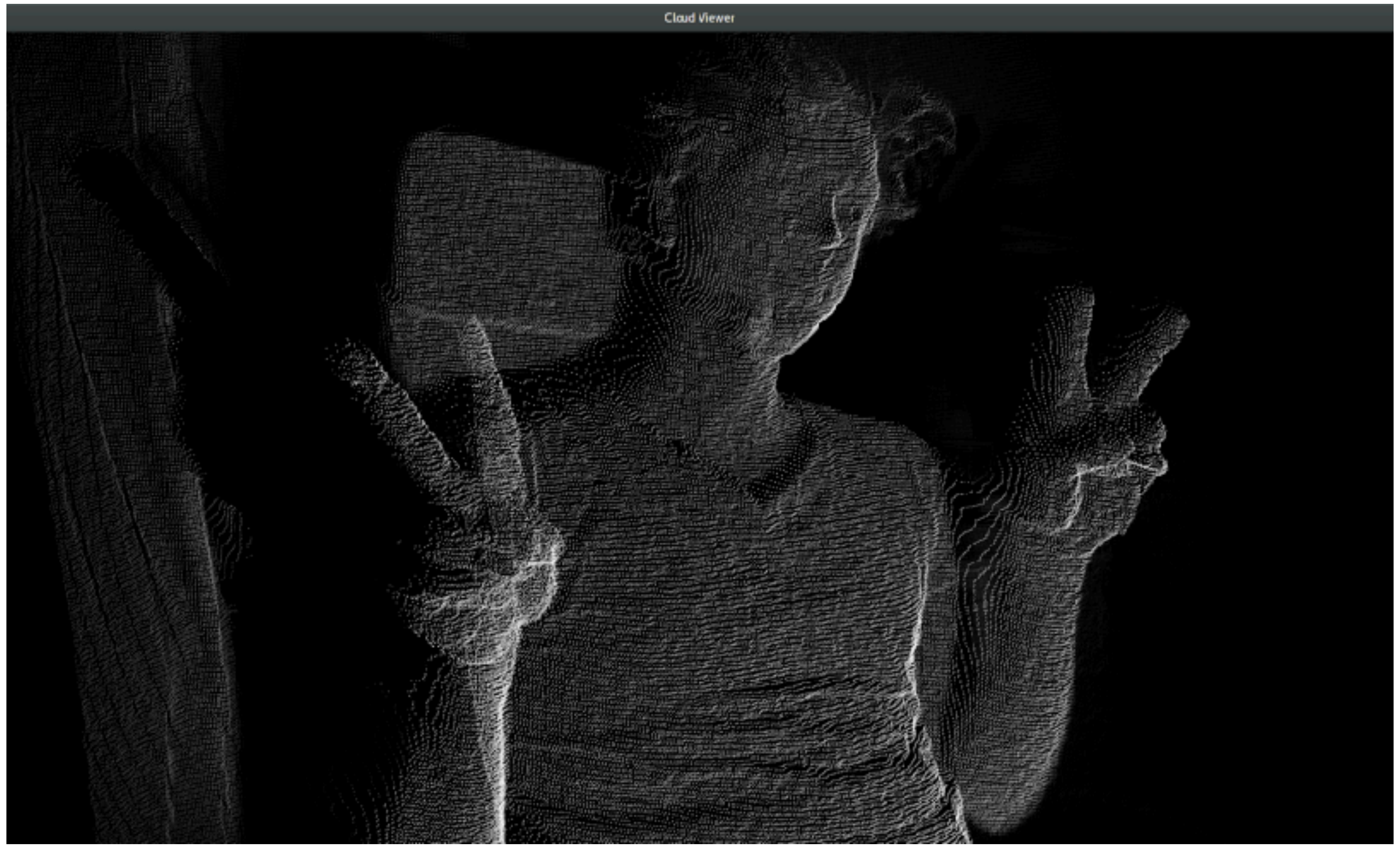
Introduction to processing

Processing?

When they started Processing in 2001, the goal was to **bring ideas and technologies** out of MIT and **into the larger world**.

They called this **sketching with code**.

Processing emerged directly from the Aesthetics and Computation Group (ACG), a research group started at the Media Lab by John Maeda in 1996.



https://medium.com/processing-foundation/a-modern-prometheus-59aed94abe85

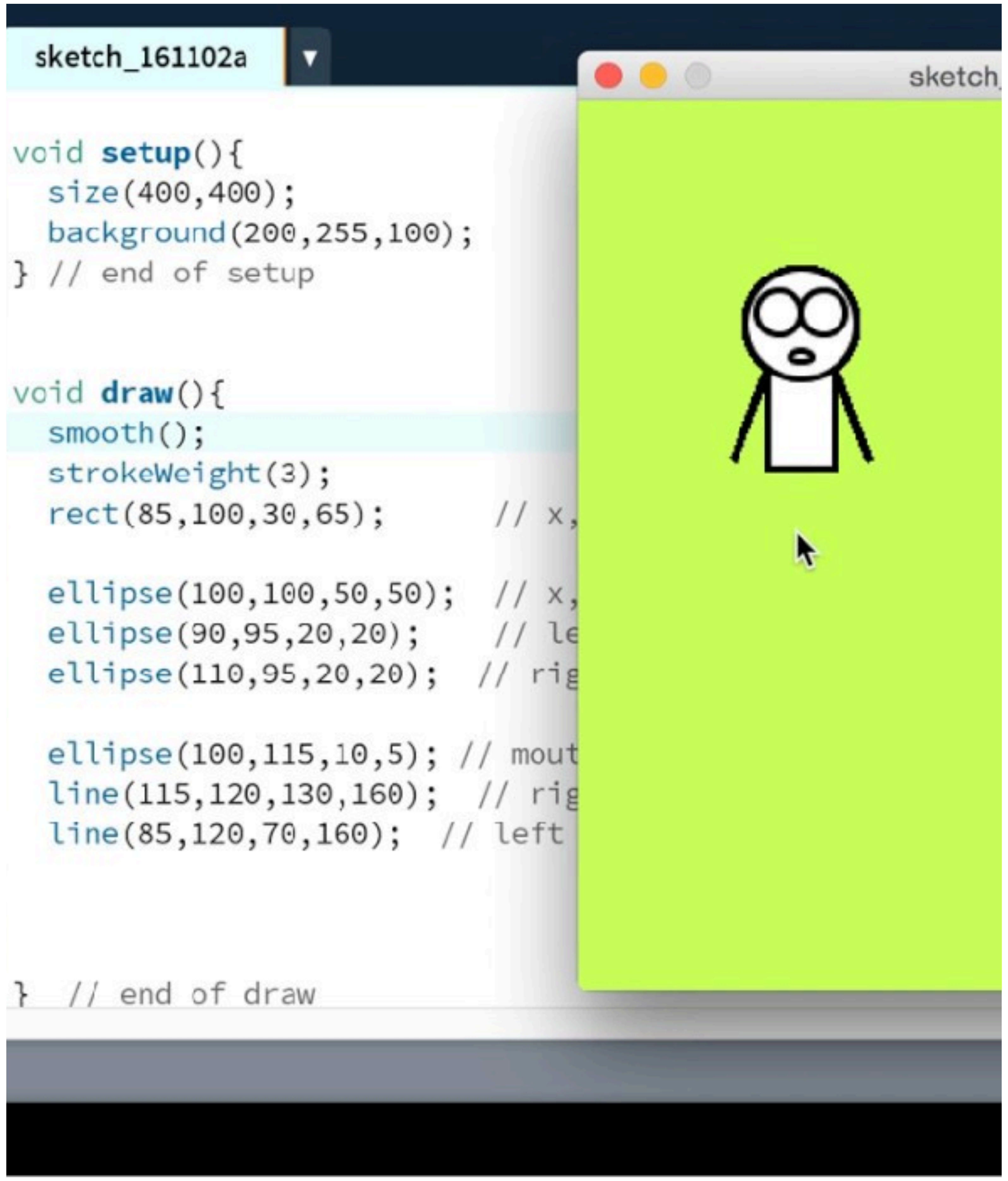
Sketching?

A Processing program is called a sketch. This is more than a change in nomenclature, **it’s a different approach to coding.**

The more traditional method is to resolve the entire plan for the software before the first line of code is written.

This approach can work well for well-defined domains, but when **the goal is exploration and invention**, it prematurely cuts off possible outcomes.

Through sketching with code, unexpected paths are discovered and followed. Unique outcomes often emerge through the process.



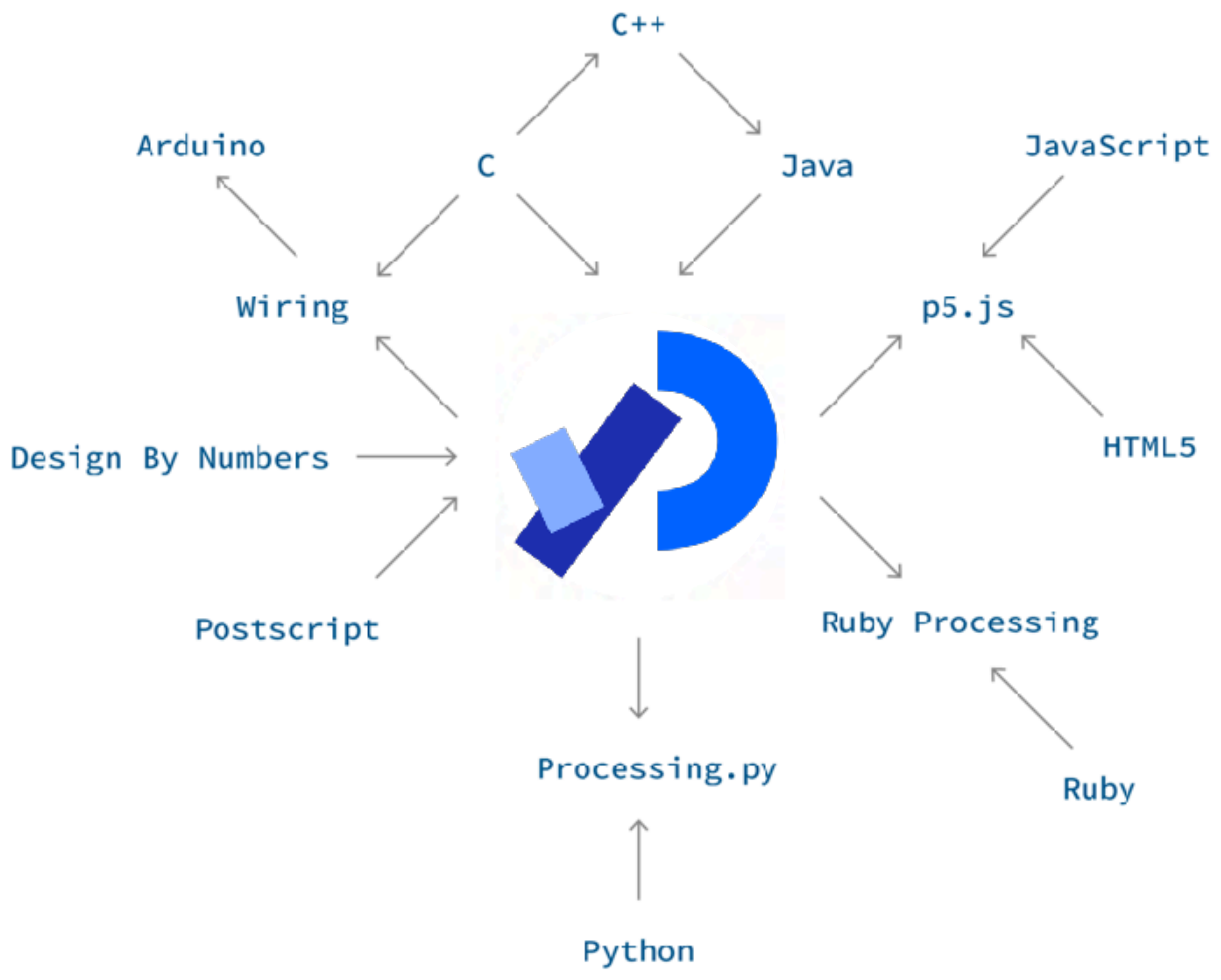
```
void setup(){
  size(400,400);
  background(200,255,100);
} // end of setup


void draw(){
  smooth();
  strokeWeight(3);
  rect(85,100,30,65);      // x,

  ellipse(100,100,50,50);  // x,
  ellipse(90,95,20,20);    // le
  ellipse(110,95,20,20);  // rig

  ellipse(100,115,10,5); // mout
  line(115,120,130,160);  // rig
  line(85,120,70,160);  // left



}  // end of draw
```

What’s for?

The original mission of Processing was to create software that made learning to code accessible for visual people (designers, artists, architects) and to help a more technical audience work fluidly with graphics.

From the original Processing software, the Foundation is now supporting a range of different projects.

The **p5.js** project is a JavaScript reimagining of Processing within the context of contemporary web browsers.
Processing.py it’s now a Mode for the Processing 3 editor. Additionally, **Processing for Android** as a Mode for Processing 3, Processing 3 running well on **Raspberry Pi** and CHIP hardware, and there is a library to read and write directly to the I/O pins.



Download

_On **Windows**, you'll have a .zip file.

Double-click it, and drag the folder inside to a location on your hard disk. It could be Program Files or simply the desktop, but the important thing is for the processing folder to be pulled out of that .zip file. Then double-click processing.exe to start.

_The **Mac OS X** version is also a .zip file.

Double-click it and drag the Processing icon to the Applications folder.
If you're using someone else's machine and can't modify the Applications folder, just drag the application to the desktop. Then double-click the Processing icon to start.

https://processing.org/download/

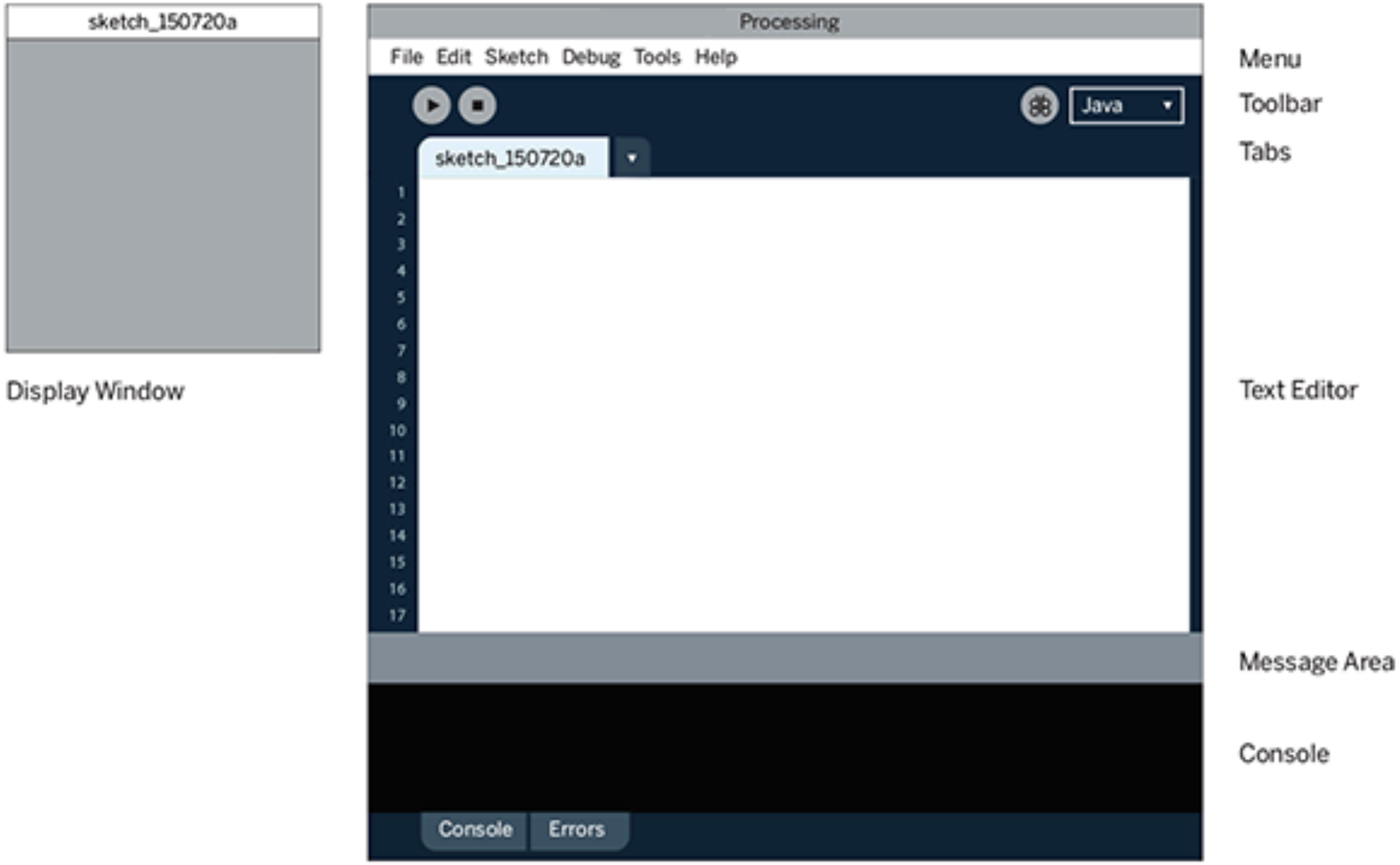
Download

Download

This Git with slides and examples

https://github.com/giulioriot/digitalArt

Hello, I’m Processing



How it works?

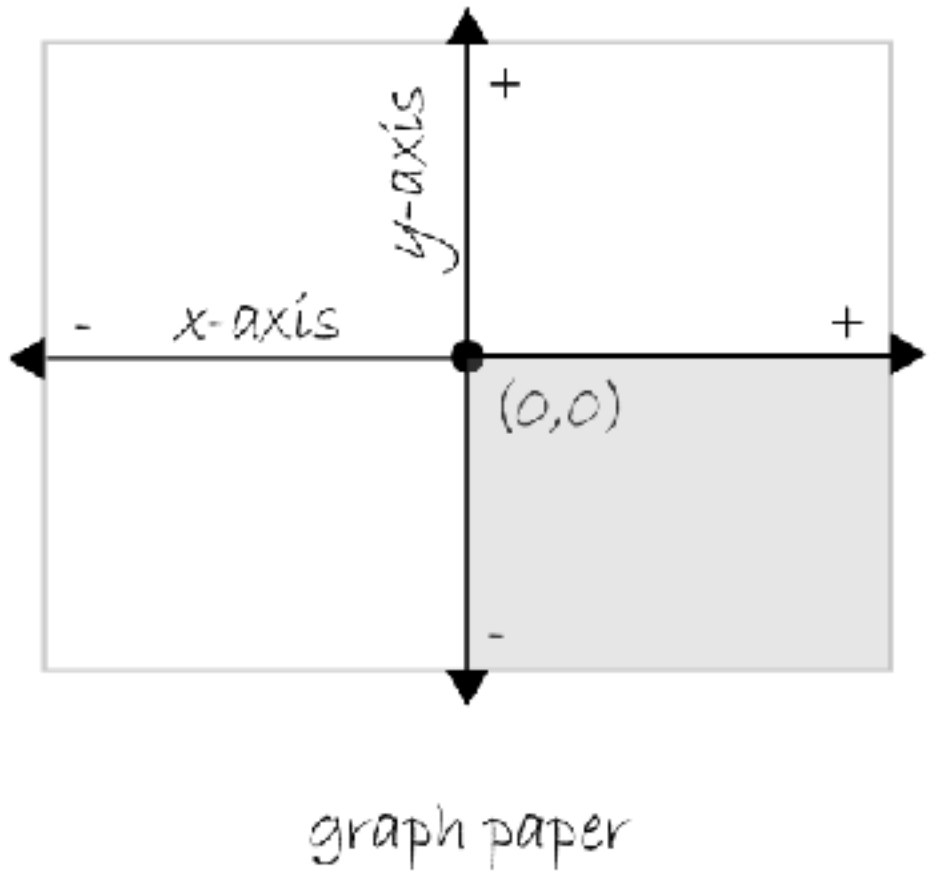
A journey of a thousand miles begins with a single step.

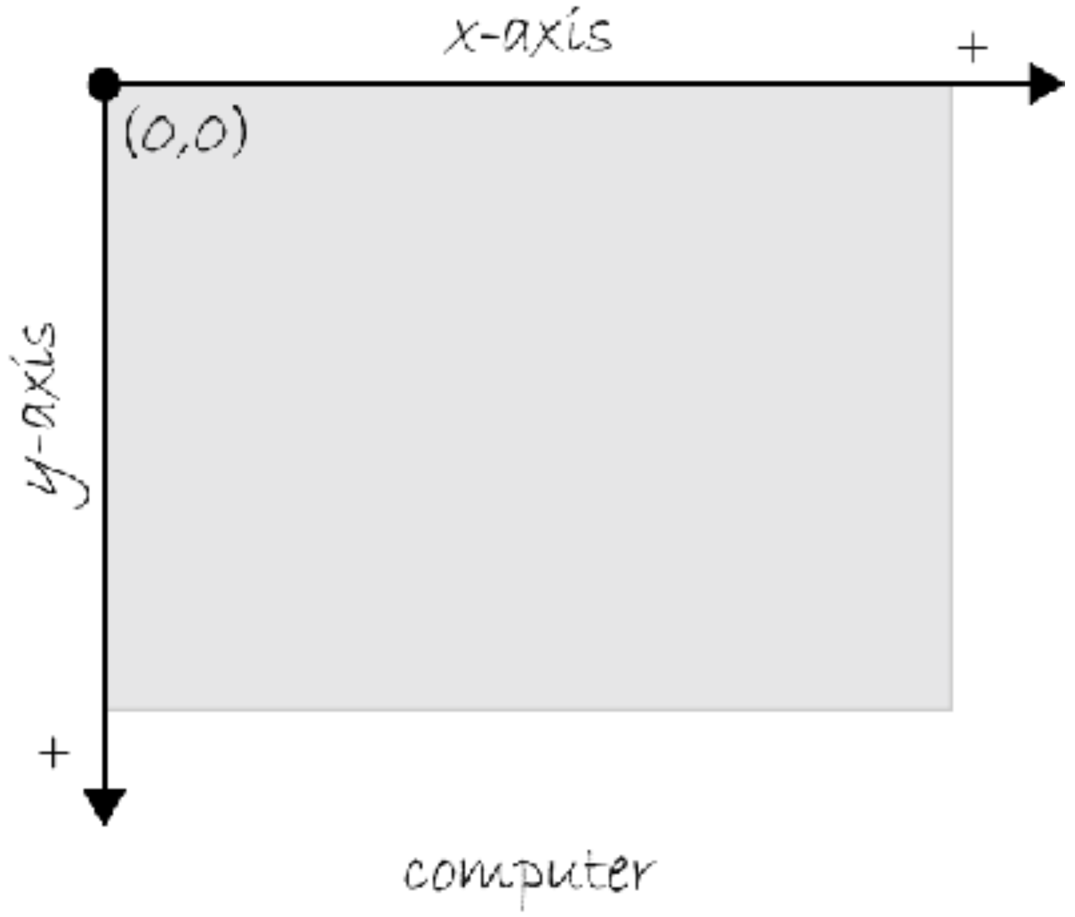
—Lao-tzu

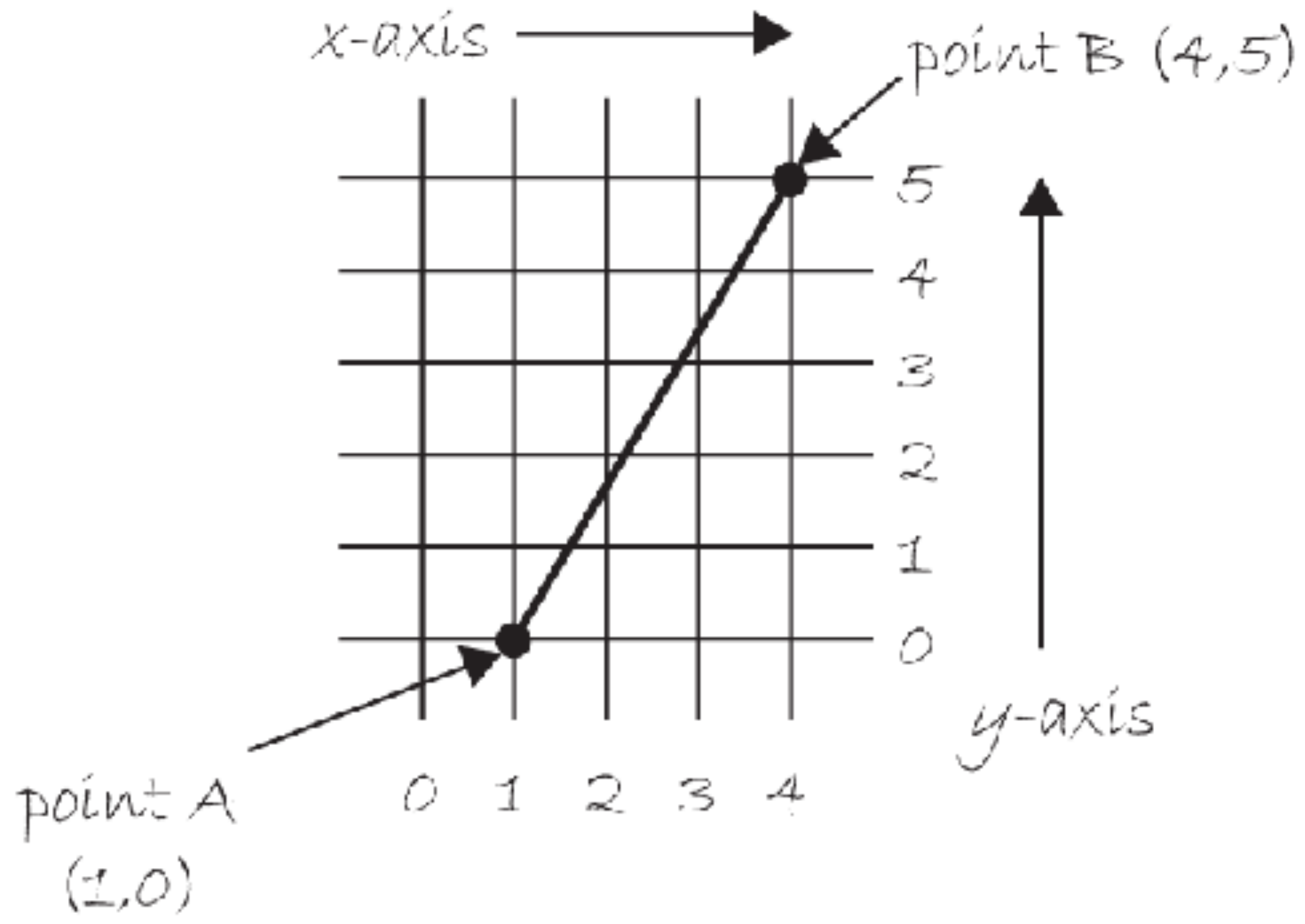
PIXELS

Digital is made by pixels, when you have to create something on a display you have to specify where you want it…

Computers thinks by pixels







This figure shows a line between point A (1,0) and point B (4,5). If you wanted to direct a friend of yours to draw that same line, you would say "draw a line from the point one-zero to the point four-five, please."

How it works?

'I try to apply colors like words that shape poems, like notes that shape music'

—Joan Miró

COLORS

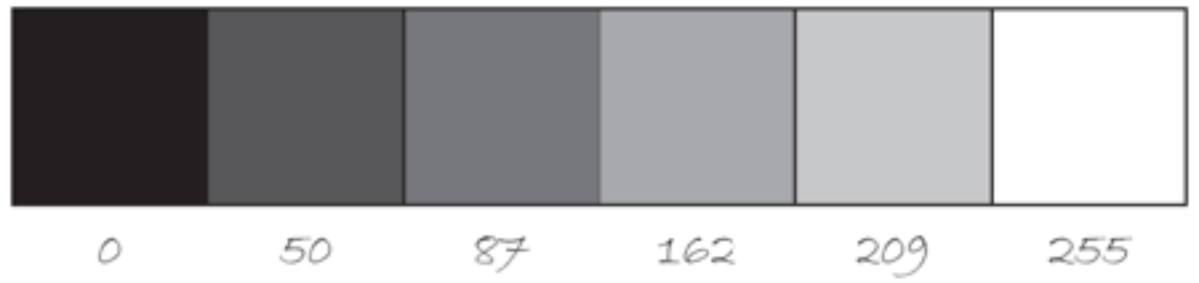
Color is defined with a range of numbers.

The simplest case: *black and white* or *grayscale*.

To specify a value for grayscale, use the following:

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.



(255, 0, 0)	(0, 255, 0)	(0, 0, 255)
(0, 255, 255)	(255, 0, 255)	(255, 255, 0)
(0, 0, 0)	(255, 255, 255)	(127, 127, 127)

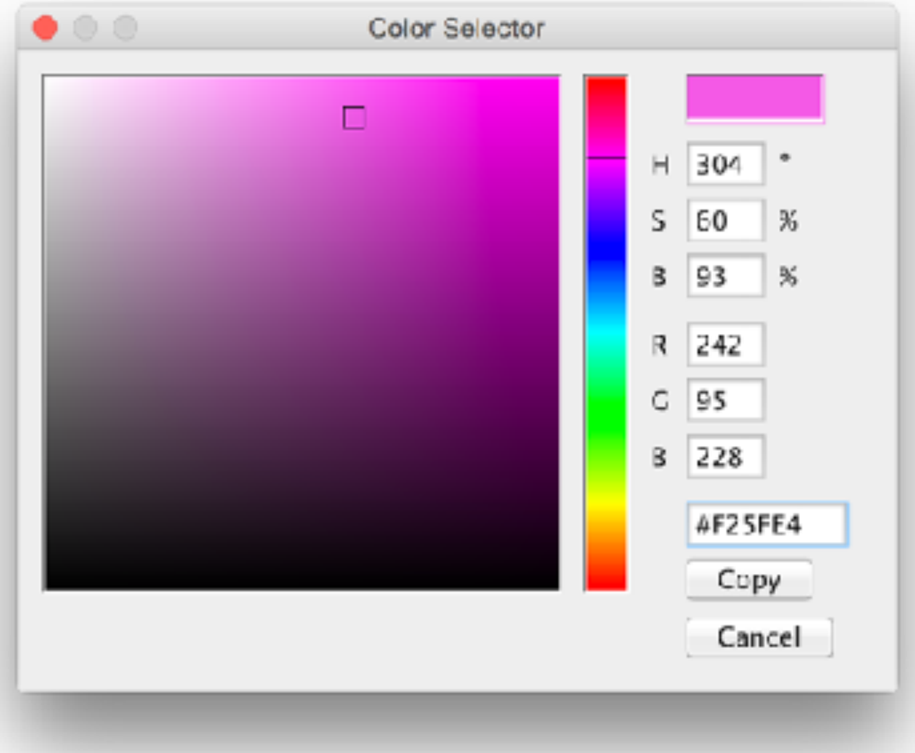
```
fill(0);
fill(255,0,0);
fill(0,255,0);
fill(0,0,255);
```

Syntax

```
fill(rgb)
fill(rgb, alpha)
fill(gray)
fill(gray, alpha)
fill(v1, v2, v3)
fill(v1, v2, v3, alpha)
```

Processing also has a color selector to aid in choosing colors.

Access this via “Tools” (from the menu bar) → “Color Selector.”



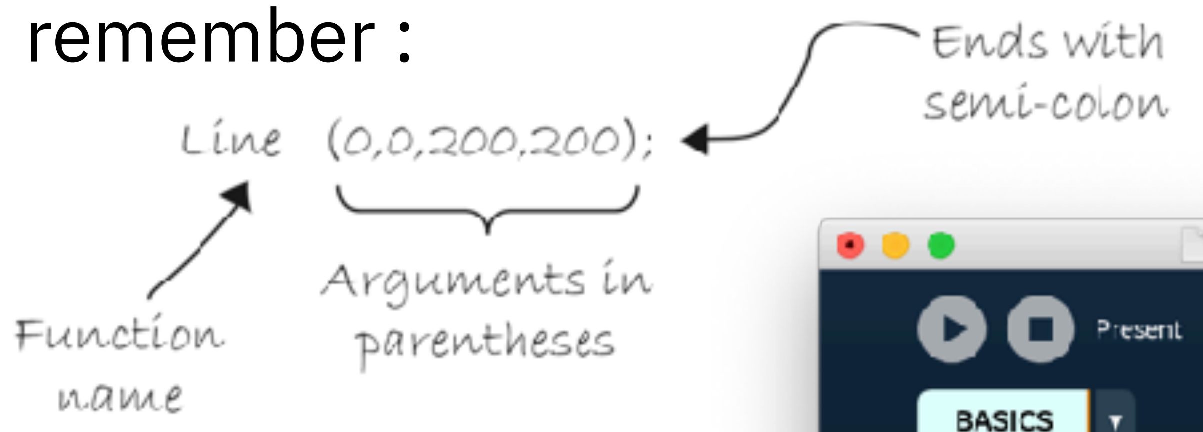
H 304 °
S 60 %
B 93 %
R 242
G 95
B 228
#F25FE4

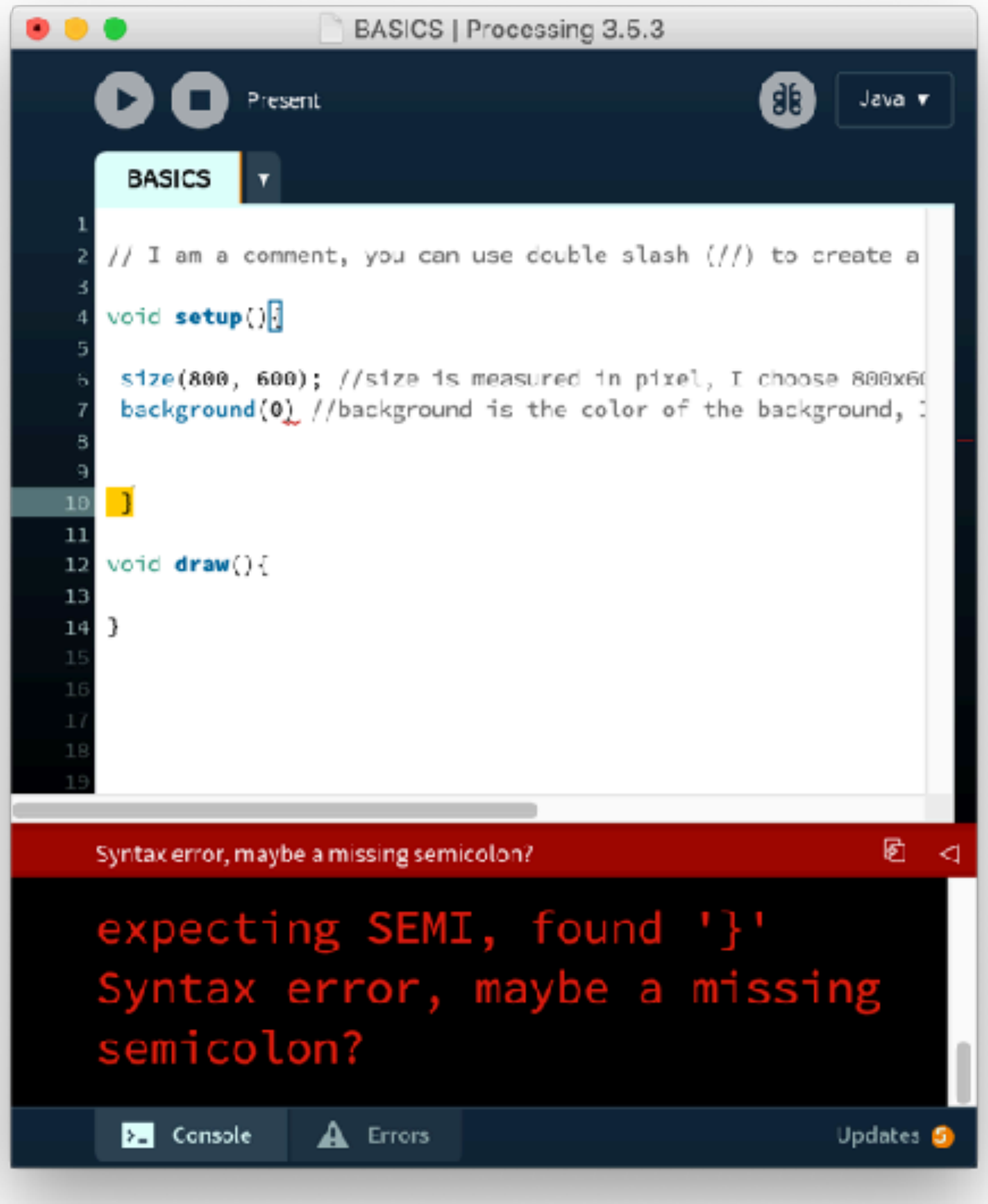
Always;

The console in the lower part shows you errors

the most frequent error is a missing semicolon one, so

remember :





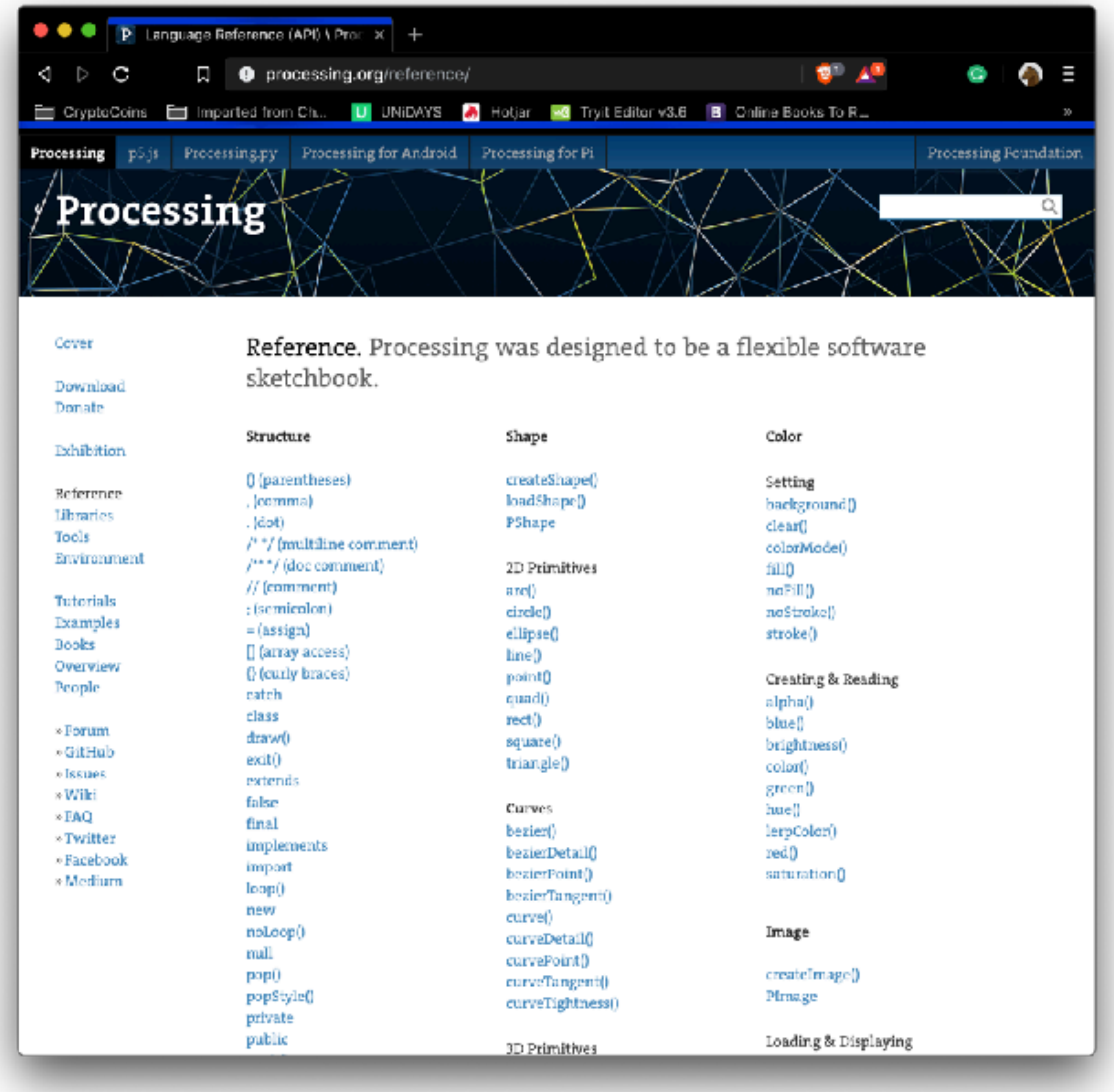
```
// I am a comment, you can use double slash (//) to create a

void setup(){

 size(800, 600); //size is measured in pixel, I choose 800x60
 background(0) //background is the color of the background, I


}

void draw(){

}
```

Syntax error, maybe a missing semicolon?

```
expecting SEMI, found '}'
Syntax error, maybe a missing semicolon?
```

everything you need is here:

https://processing.org/reference/



Cover

Download
Donate

Exhibition

Reference
Libraries
Tools
Environment

Tutorials
Examples
Books
Overview
People

- Forum
- GitHub
- Issues
- Wiki
- FAQ
- Twitter
- Facebook
- Medium

Reference. Processing was designed to be a flexible software sketchbook.

Structure

() (parentheses)
, (comma)
. (dot)
/* */ (multiline comment)
/** */ (doc comment)
// (comment)
; (semicolon)
= (assign)
[] (array access)
{} (curly braces)
catch
class
draw()
exit()
extends
false
final
implements
import
loop()
new
noLoop()
null
pop()
popStyle()
private
public

Shape

createShape()
loadShape()
PShape

2D Primitives
arc()
circle()
ellipse()
line()
point()
quad()
rect()
square()
triangle()

Curves
bezier()
bezierDetail()
bezierPoint()
bezierTangent()
curve()
curveDetail()
curvePoint()
curveTangent()
curveTightness()

3D Primitives

Color

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

Creating & Reading
alpha()
blue()
brightness()
color()
green()
hue()
lerpColor()
red()
saturation()

Image

createImage()
PImage

Loading & Displaying

Let’s create a sketch

The keyword **void** indicates a function with no value.

If you don't know what a function is...

void setup(){

is the space in which I setup my sketch

Everything you write in the SETUP will run just once

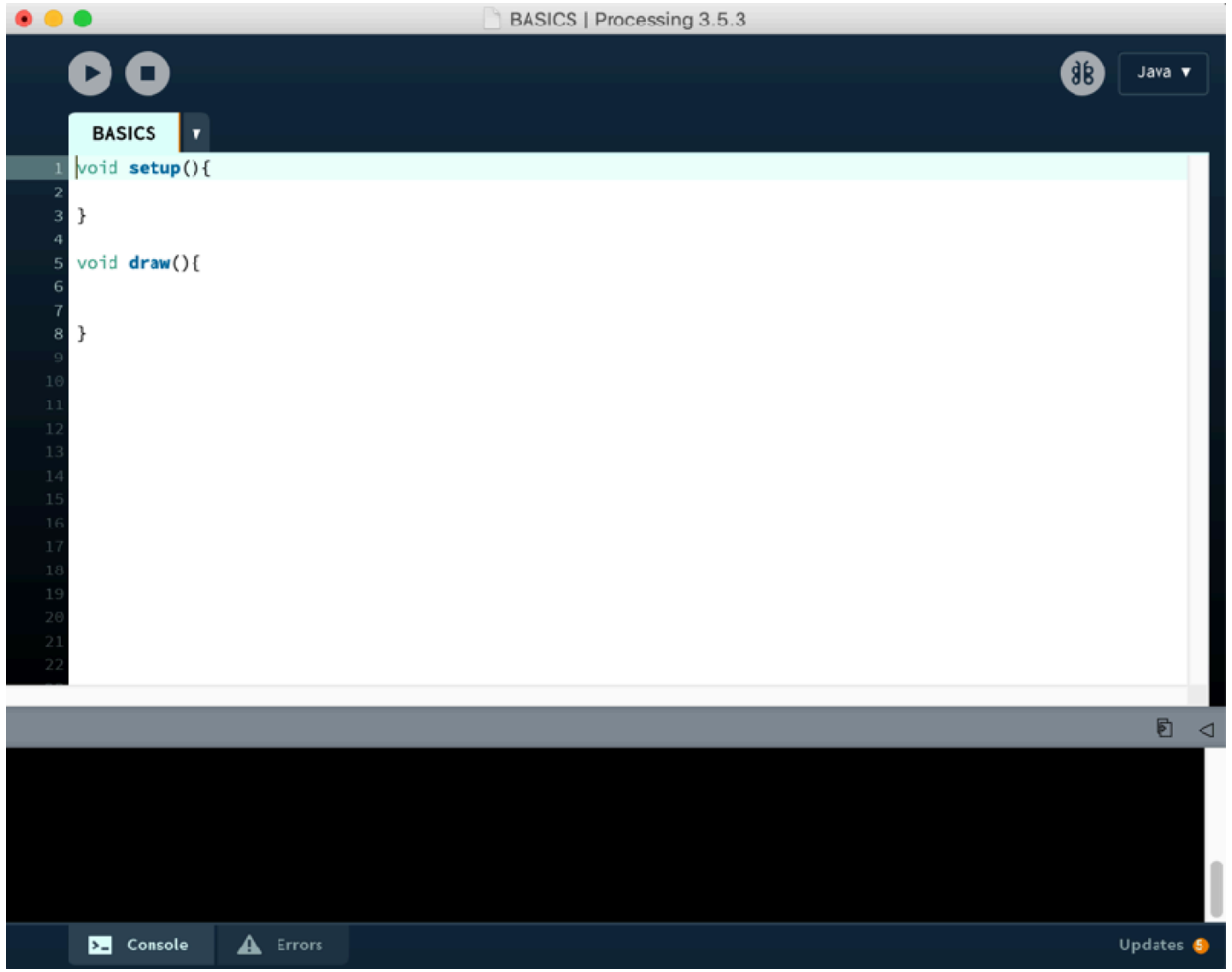
}

void draw(){

is the space in which I draw in my sketch

Everything you write in the DRAW, will run in an infinite loop

}



```
void setup(){

}

void draw(){


}
```

Basics

POLITECNICO
MILANO 1863

```
// open the folder of examples

sketch_1BASICS
```

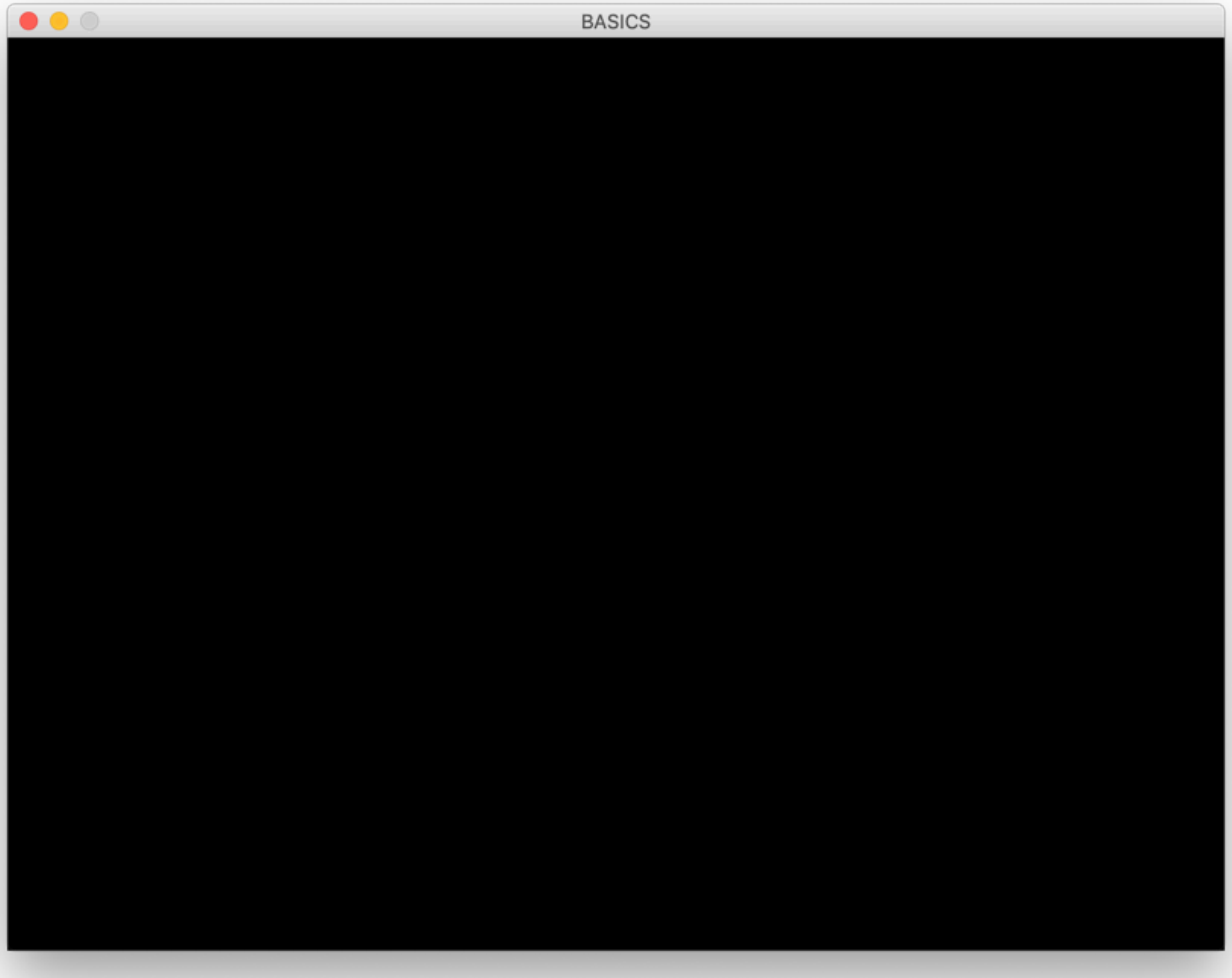
here you find

A black blank page

our hello world

tips:

try to change the size of the sketch and the background color at line 10 and 12



Shapes

POLITECNICO
MILANO 1863

```
// open the folder of examples

sketch_2BASIC_SHAPES
```

here you find

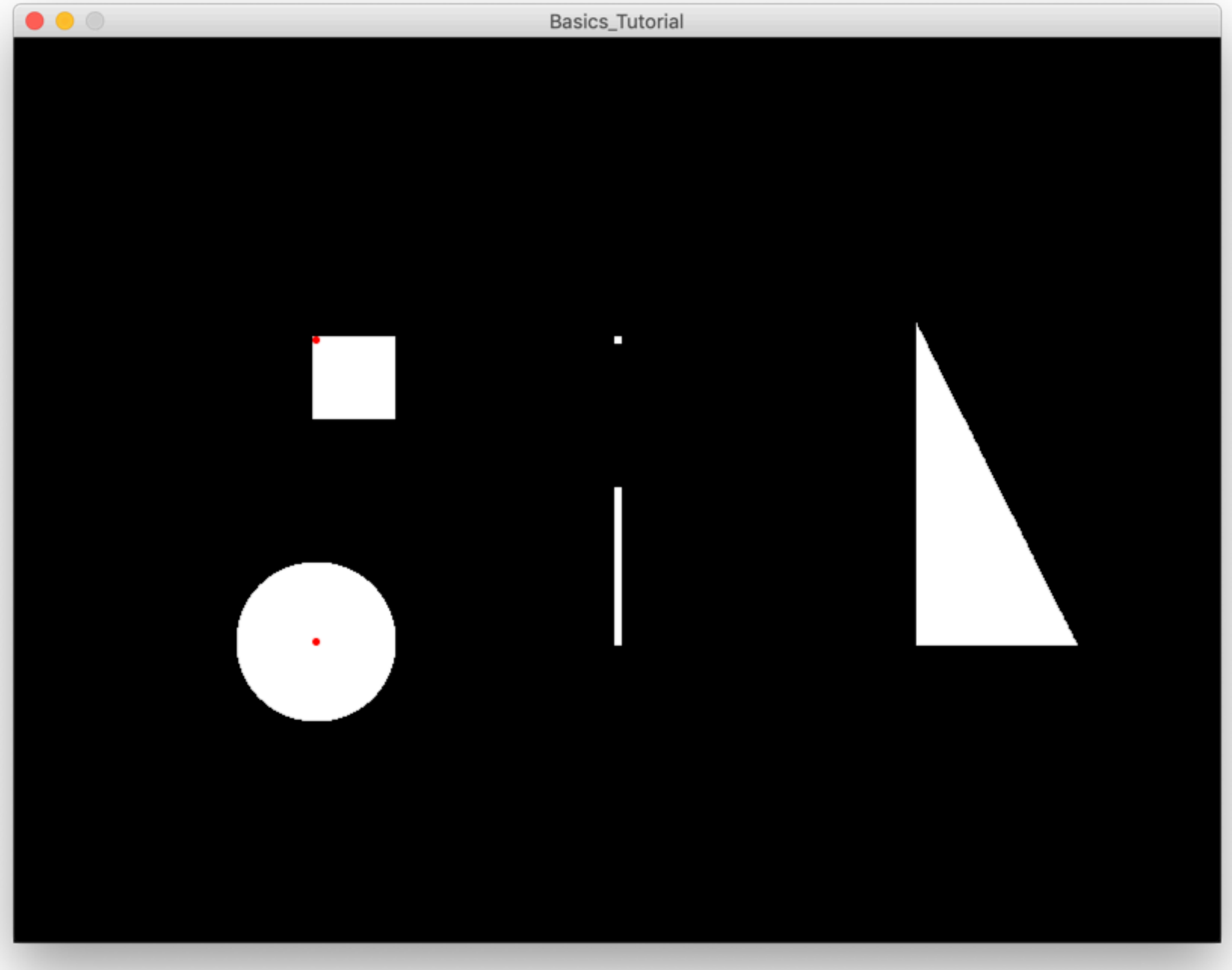
Primitive shapes

to understand how to draw by code

tips:

change from a grayscale color to rgb at lines 8 and 14

try to modify the triangle shape at line 24



Stickman

```
// open the folder of examples
```

```
sketch_3STICK_MAN
```

here you find

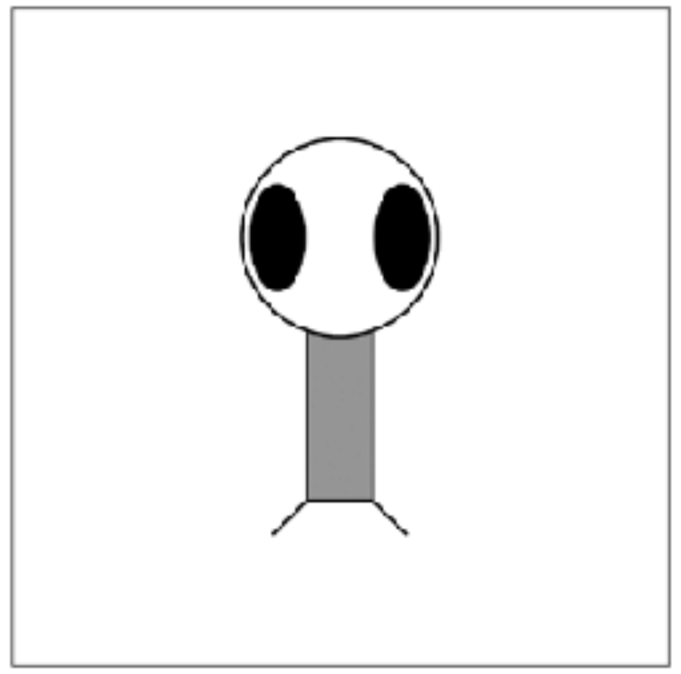
A simple stickman

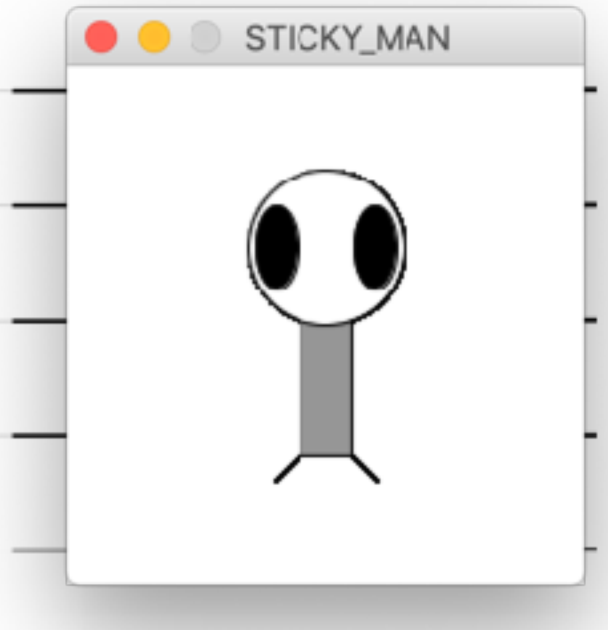
hack and modify code to obtain a personal output

`tips:`

try to add arms and details, be creative!

```
background(255);
ellipseMode(CENTER);
rectMode(CENTER);
stroke(0);
fill(150);
rect(100, 100, 20, 100);
fill(255);
ellipse(100, 70, 60, 60);
fill(0);
ellipse(81, 70, 16, 32);
ellipse(119, 70, 16, 32);
stroke(0);
line(90, 150, 80, 160);
line(110, 150, 120, 160);
```

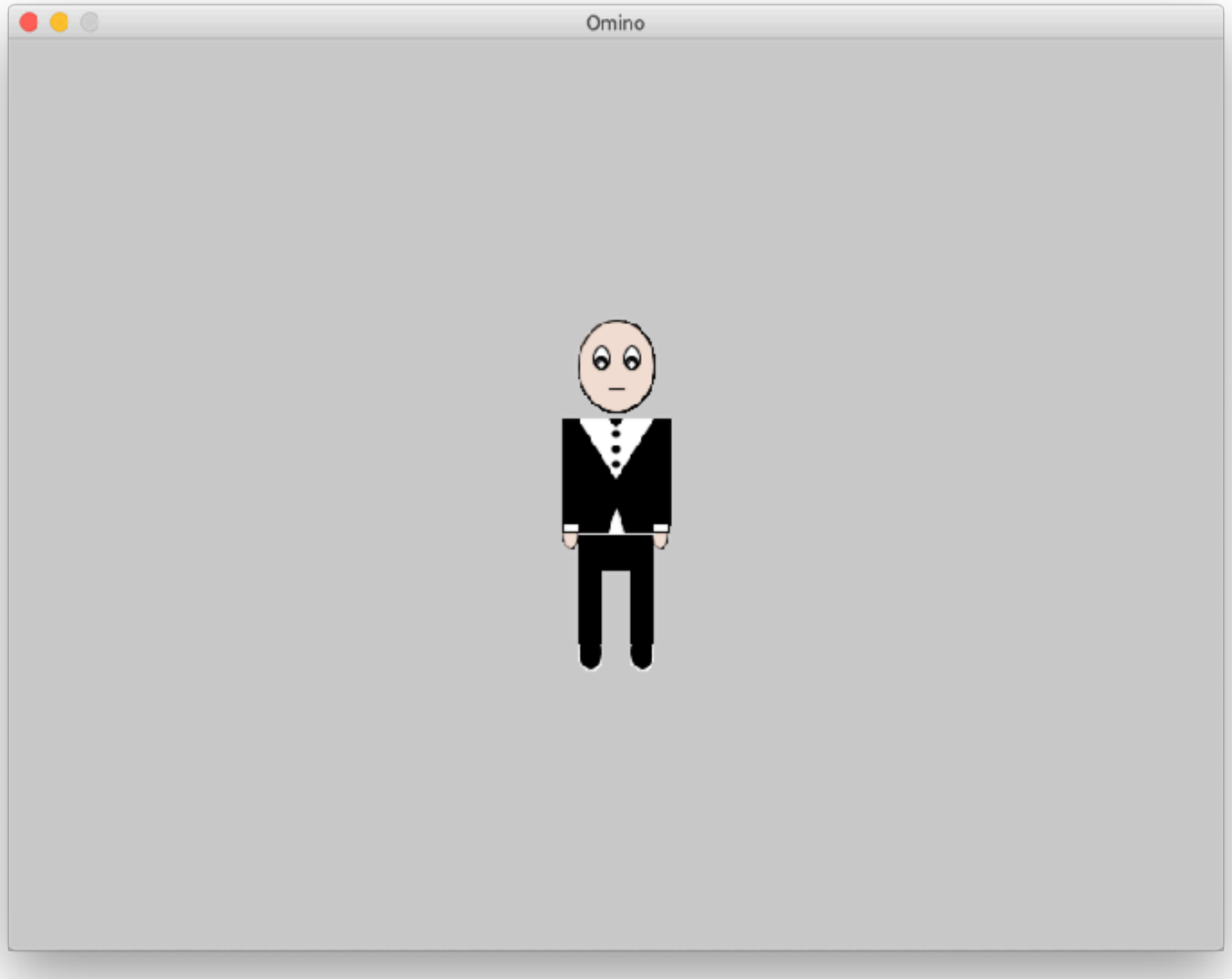


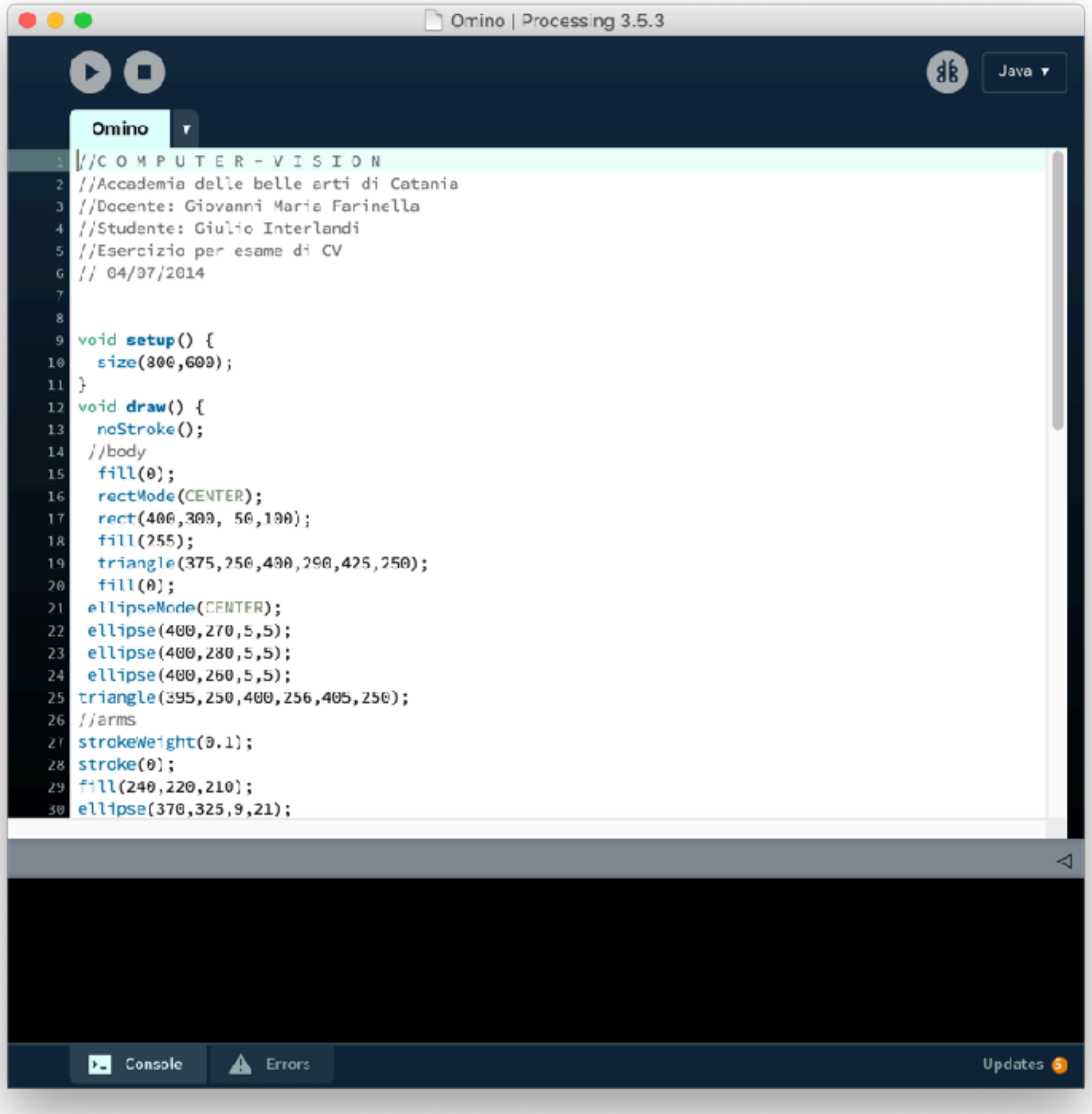


Be creative!

Let’s create your own

that was my exercise some years ago



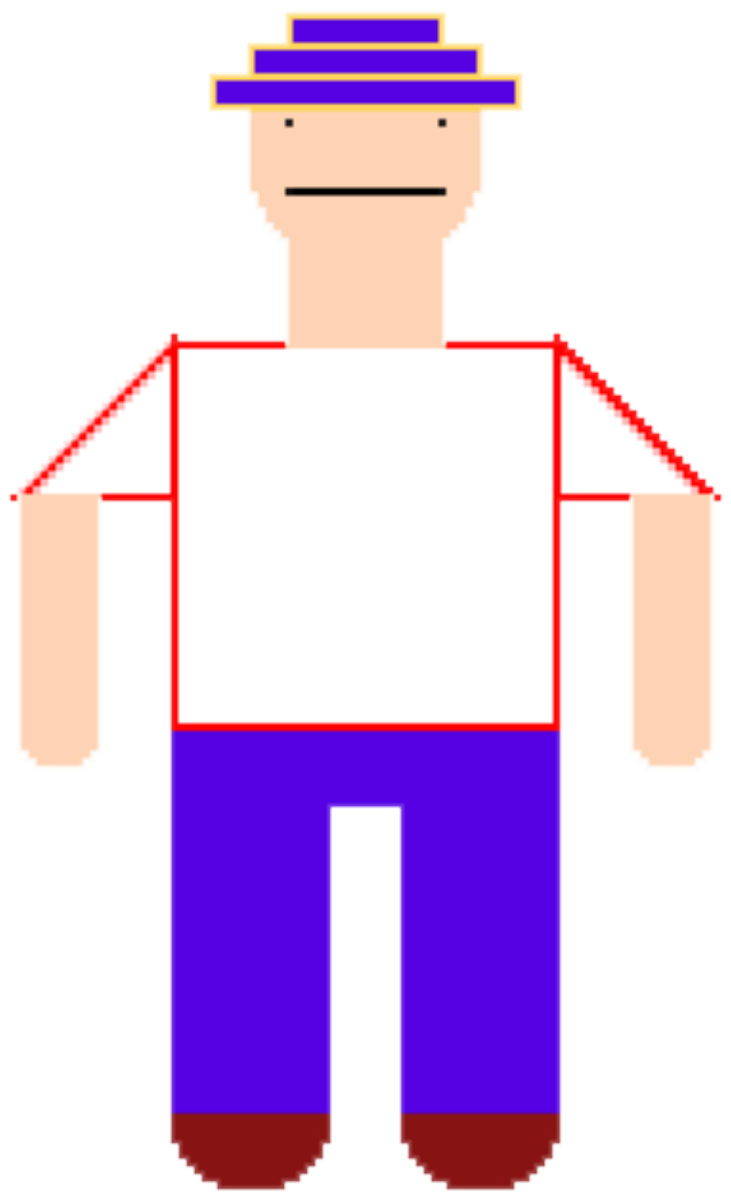


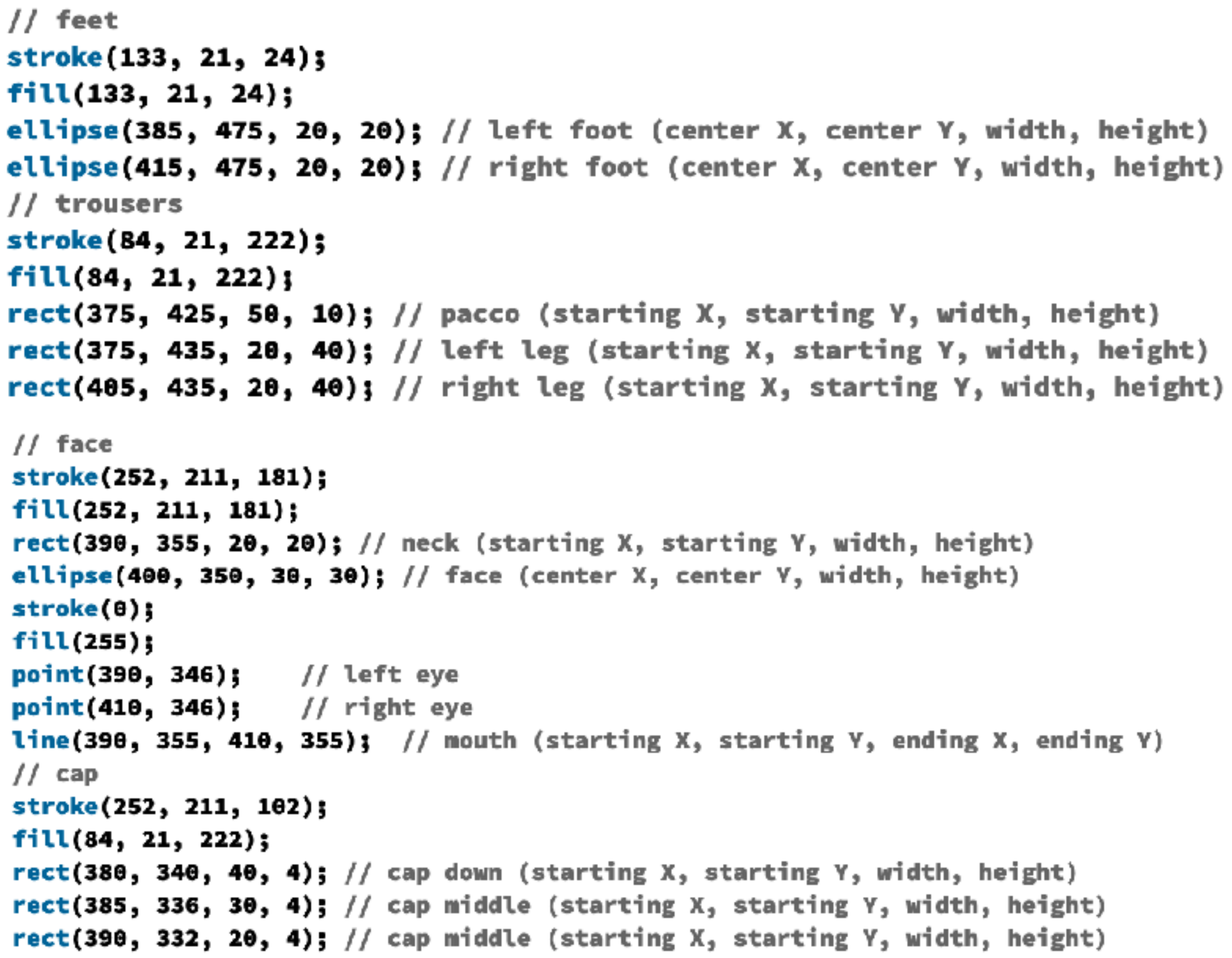
```
//C O M P U T E R - V I S I O N
//Accademia delle belle arti di Catania
//Docente: Giovanni Maria Farinella
//Studente: Giulio Interlandi
//Esercizio per esame di CV
// 04/07/2014


void setup() {
  size(800,600);
}
void draw() {
  noStroke();
 //body
  fill(0);
  rectMode(CENTER);
  rect(400,300, 50,100);
  fill(255);
  triangle(375,250,400,290,425,250);
  fill(0);
 ellipseMode(CENTER);
 ellipse(400,270,5,5);
 ellipse(400,280,5,5);
 ellipse(400,260,5,5);
triangle(395,250,400,256,405,250);
//arms
strokeWeight(0.1);
stroke(0);
fill(240,220,210);
ellipse(370,325,9,21);
```

Be creative!

Let’s create your own

this is a funny sketch done by a student last year.





```
// feet
stroke(133, 21, 24);
fill(133, 21, 24);
ellipse(385, 475, 20, 20); // left foot (center X, center Y, width, height)
ellipse(415, 475, 20, 20); // right foot (center X, center Y, width, height)
// trousers
stroke(84, 21, 222);
fill(84, 21, 222);
rect(375, 425, 50, 10); // pacco (starting X, starting Y, width, height)
rect(375, 435, 20, 40); // left leg (starting X, starting Y, width, height)
rect(405, 435, 20, 40); // right leg (starting X, starting Y, width, height)

// face
stroke(252, 211, 181);
fill(252, 211, 181);
rect(390, 355, 20, 20); // neck (starting X, starting Y, width, height)
ellipse(400, 350, 30, 30); // face (center X, center Y, width, height)
stroke(0);
fill(255);
point(390, 346);    // left eye
point(410, 346);    // right eye
line(390, 355, 410, 355);  // mouth (starting X, starting Y, ending X, ending Y)
// cap
stroke(252, 211, 102);
fill(84, 21, 222);
rect(380, 340, 40, 4); // cap down (starting X, starting Y, width, height)
rect(385, 336, 30, 4); // cap middle (starting X, starting Y, width, height)
rect(390, 332, 20, 4); // cap middle (starting X, starting Y, width, height)
```

Credits: Luca Ghezzi

Let’s create a sketch

POLITECNICO
MILANO 1863

```
// open the folder of examples
```

```
sketch_4DRAWING TOOL
```

here you find

Drawing principles

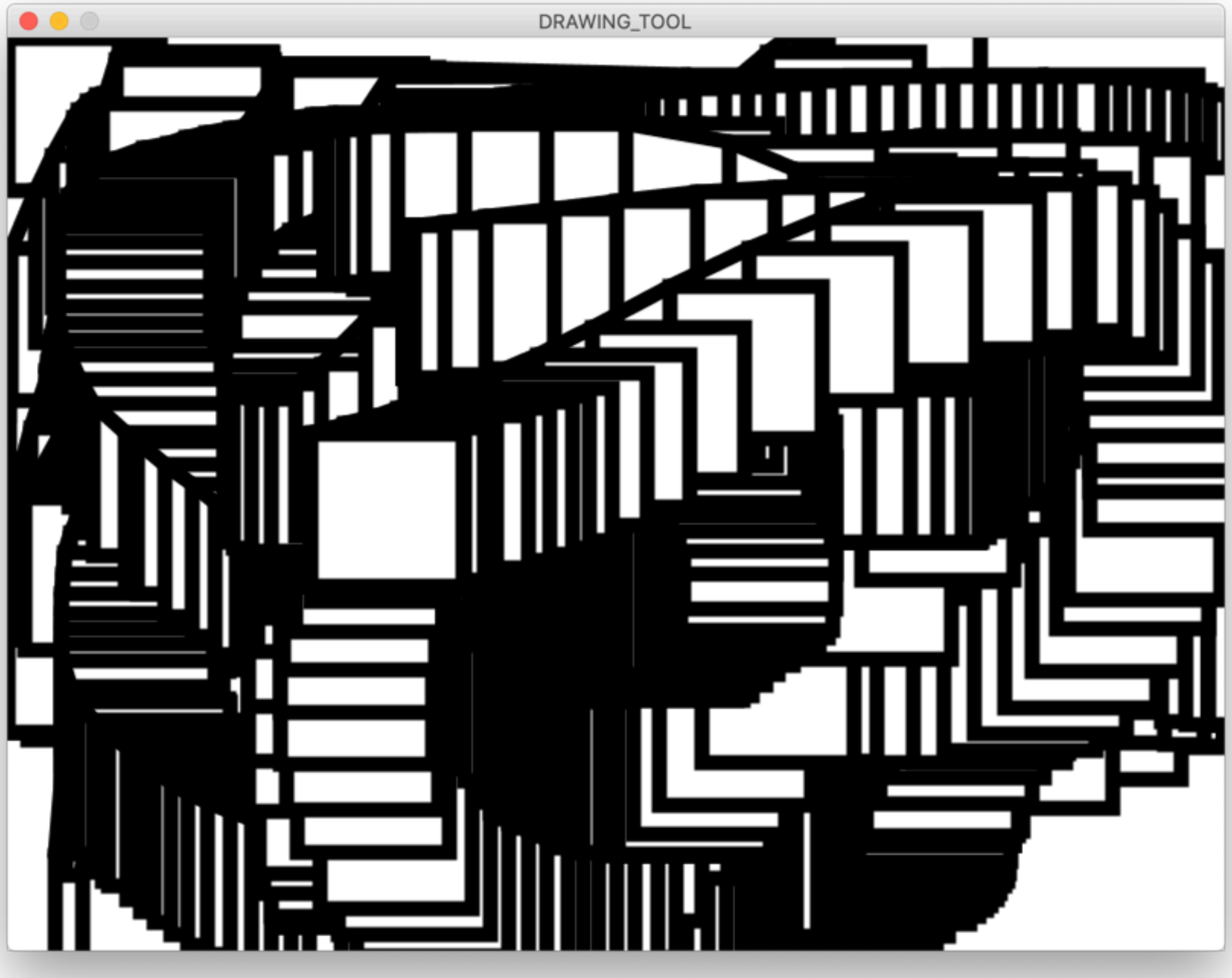
to understand:

•how to connect and draw by mouse input

• if (key pressed) case

`tips:`

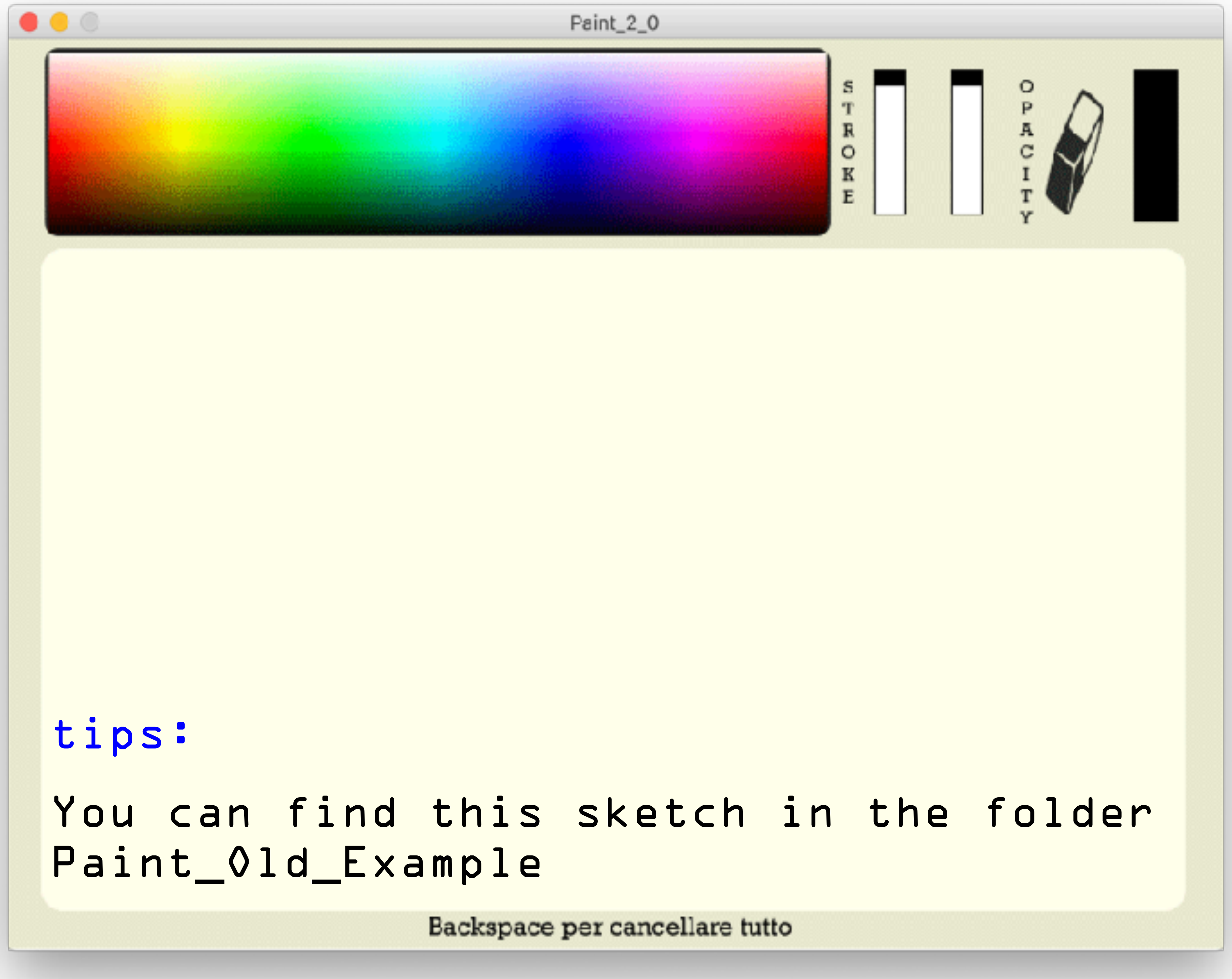
delete or comment the background at line 16 and the rectangle at line 17, uncomment line 19 and the “if” at line 22 to 24



Be creative!

Let’s create your own

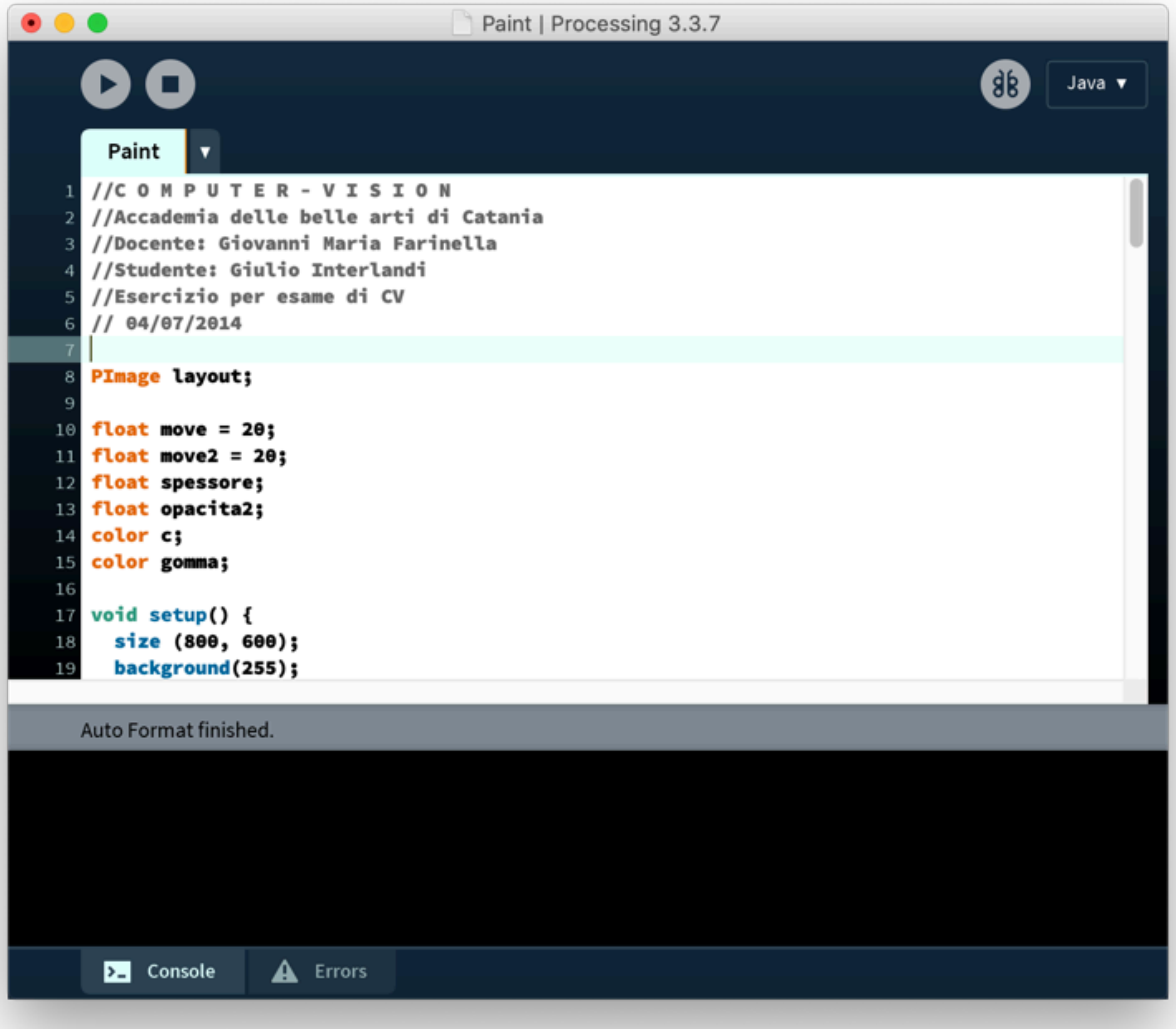
that was my exercise some years ago



tips:

You can find this sketch in the folder Paint_Old_Example

Backspace per cancellare tutto



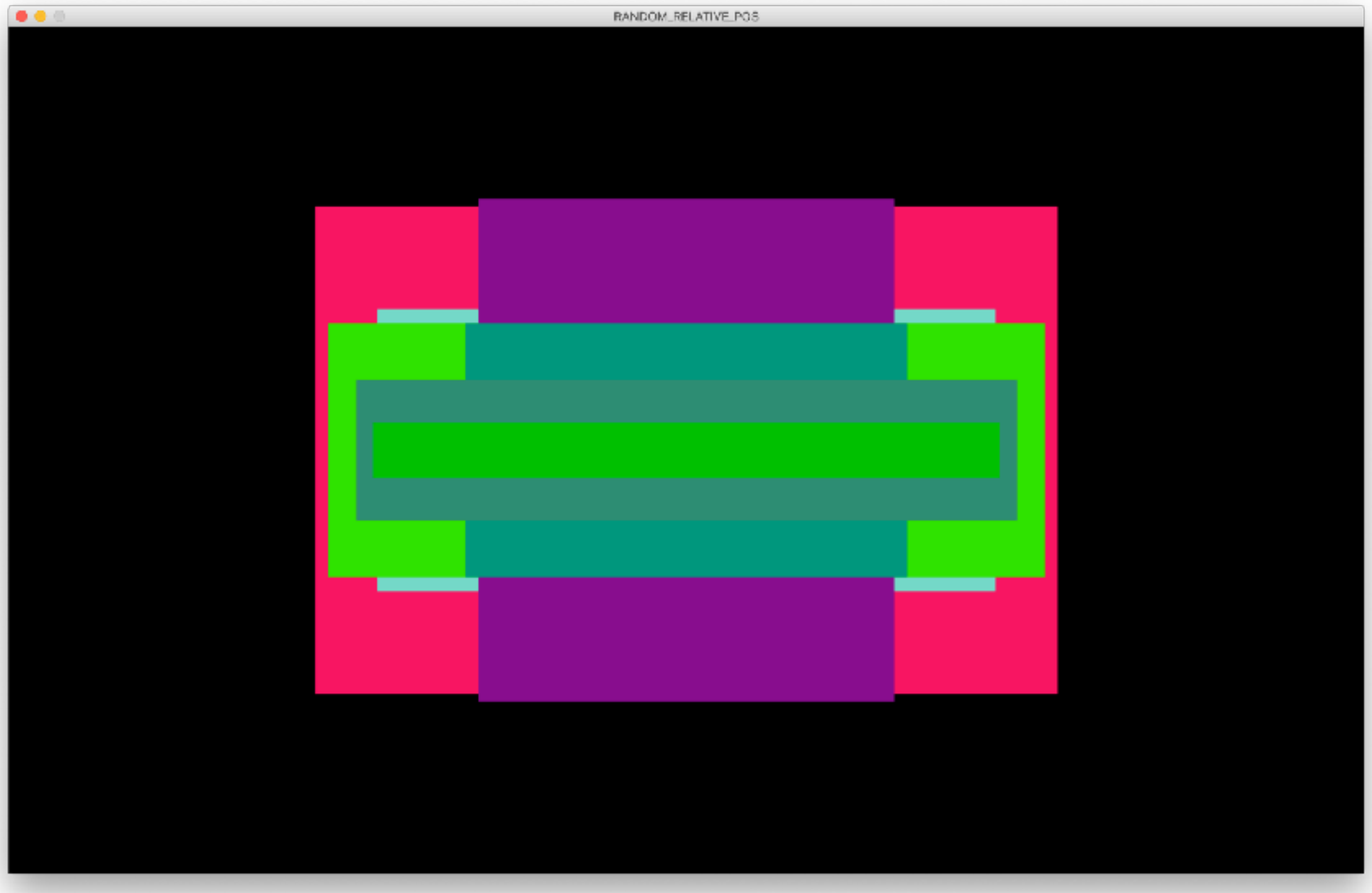
```
//C O M P U T E R - V I S I O N
//Accademia delle belle arti di Catania
//Docente: Giovanni Maria Farinella
//Studente: Giulio Interlandi
//Esercizio per esame di CV
// 04/07/2014

PImage layout;

float move = 20;
float move2 = 20;
float spessore;
float opacita2;
color c;
color gomma;

void setup() {
  size (800, 600);
  background(255);
```

```
Auto Format finished.
```

Let’s create a sketch

```
// open the folder of examples
```

```
sketch_5RANDOM_RELATIVE
```



here you find

Random principles and coordinates

to understand randomness and positioning shapes without writing pixel

tips:

try to comment the filter at line 15 and delay at line 16

Frieder Nake's remixes

```
// open the folder of examples
```

```
NAKE_REMIX
```

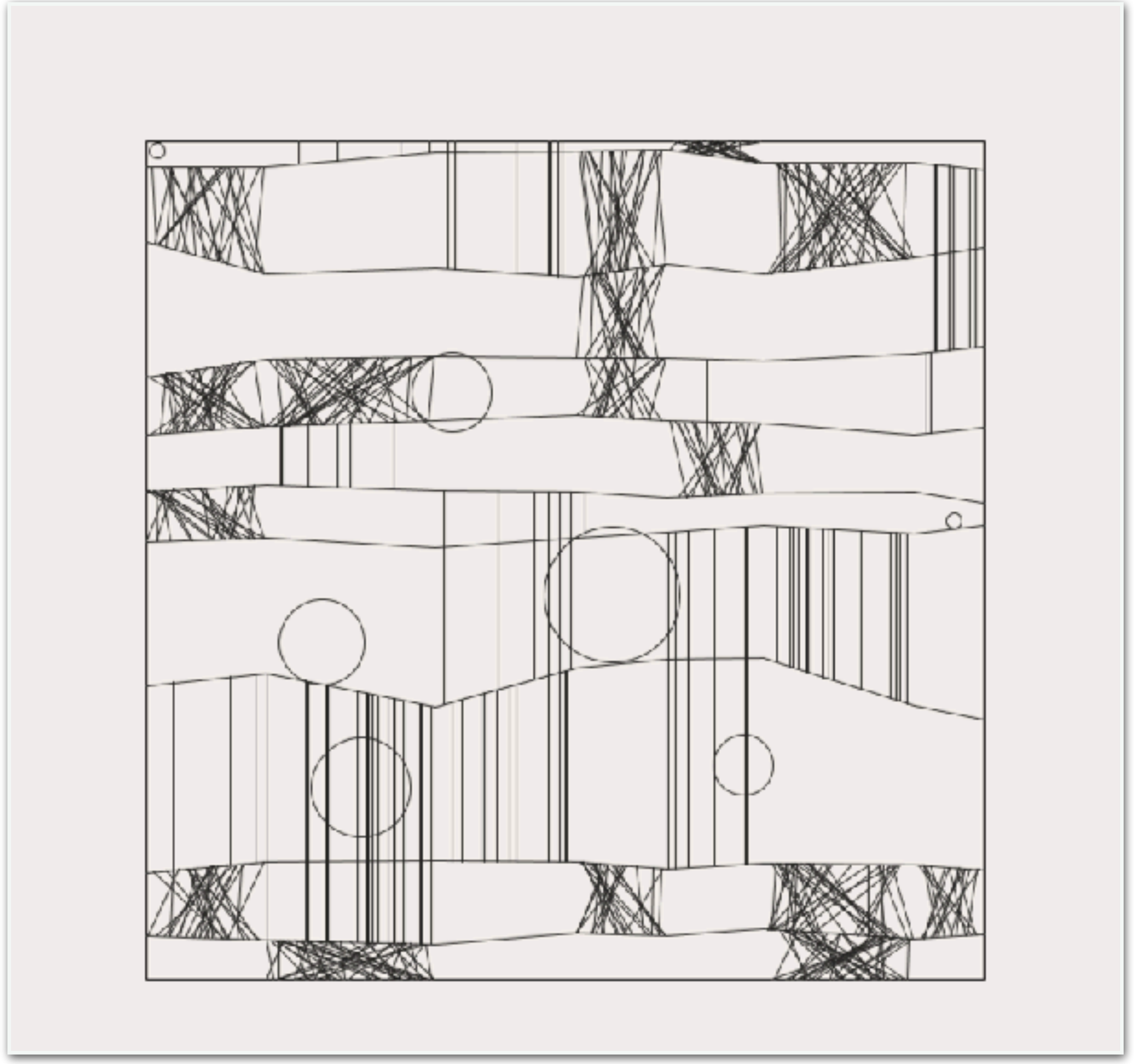
here you find

Remixes of Nake’s works and open processing platform

to understand how to manage more complex sketches and discover the open processing platform

```
tips:
```

try to understand how these sketches work, and make your own remix



Credits: Max Cura find it at Open Processing page

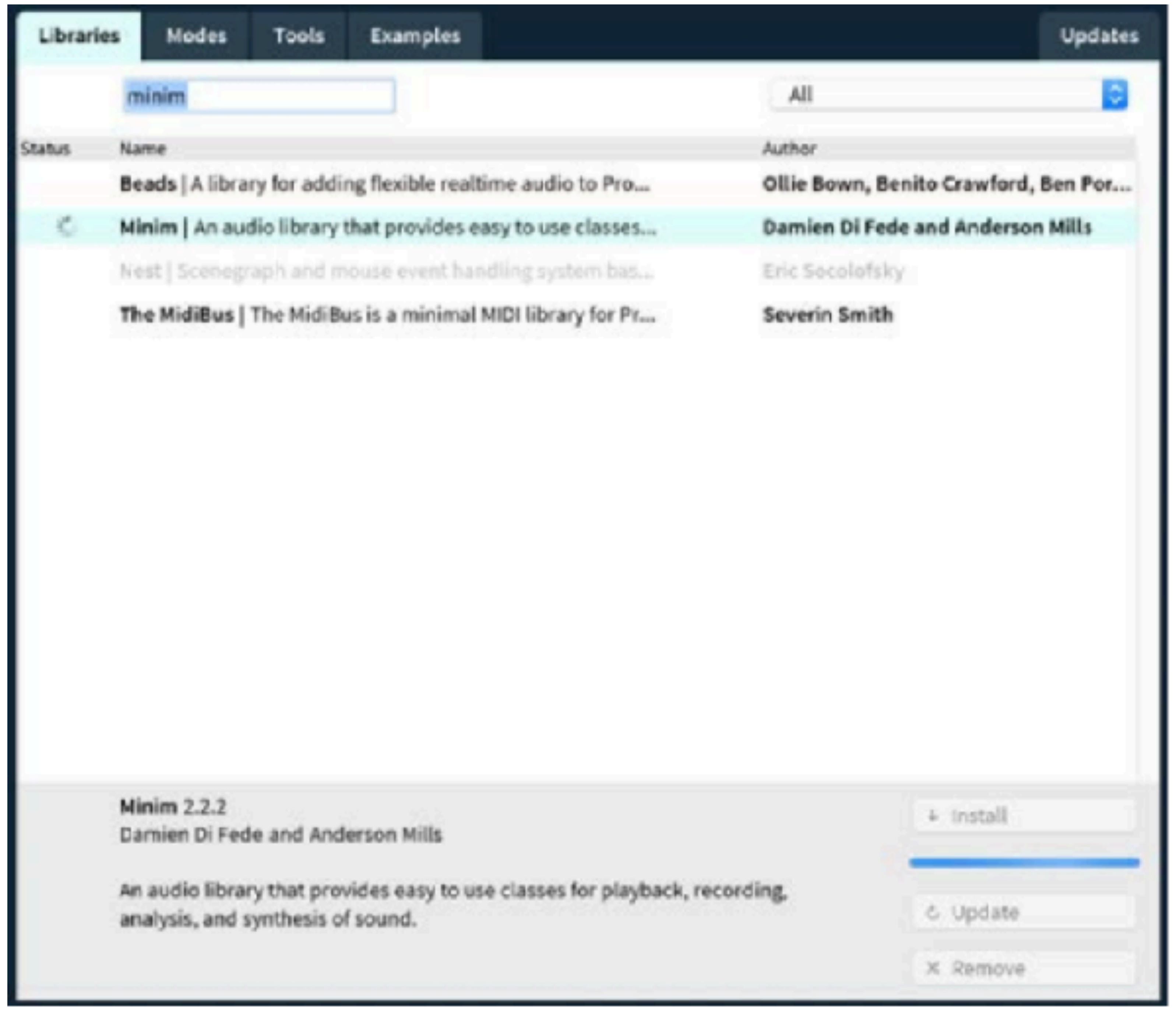
\

In order to work with sound information in Processing, we first need to **install an extension library** since Processing doesn't natively support this feature.

It can be simply installed by opening:

Sketch
➜
Import Library
➜
Add Library, typing 'minim' into the search field and then clicking on 'Install'.

```
// windows users

Minim or Sound libraries
```



Status	Name	Author
	Beads | A library for adding flexible realtime audio to Pro...	Ollie Bown, Benito Crawford, Ben Por...
	Minim | An audio library that provides easy to use classes...	Damien Di Fede and Anderson Mills
	Nest | Scenegraph and mouse event handling system bas...	Eric Socolofsky
	The MidiBus | The MidiBus is a minimal MIDI library for Pr...	Severin Smith

Minim 2.2.2
Damien Di Fede and Anderson Mills

An audio library that provides easy to use classes for playback, recording, analysis, and synthesis of sound.

You can find more in the document:

Processing-Generative_Design_Tutorial_soundmapping

\

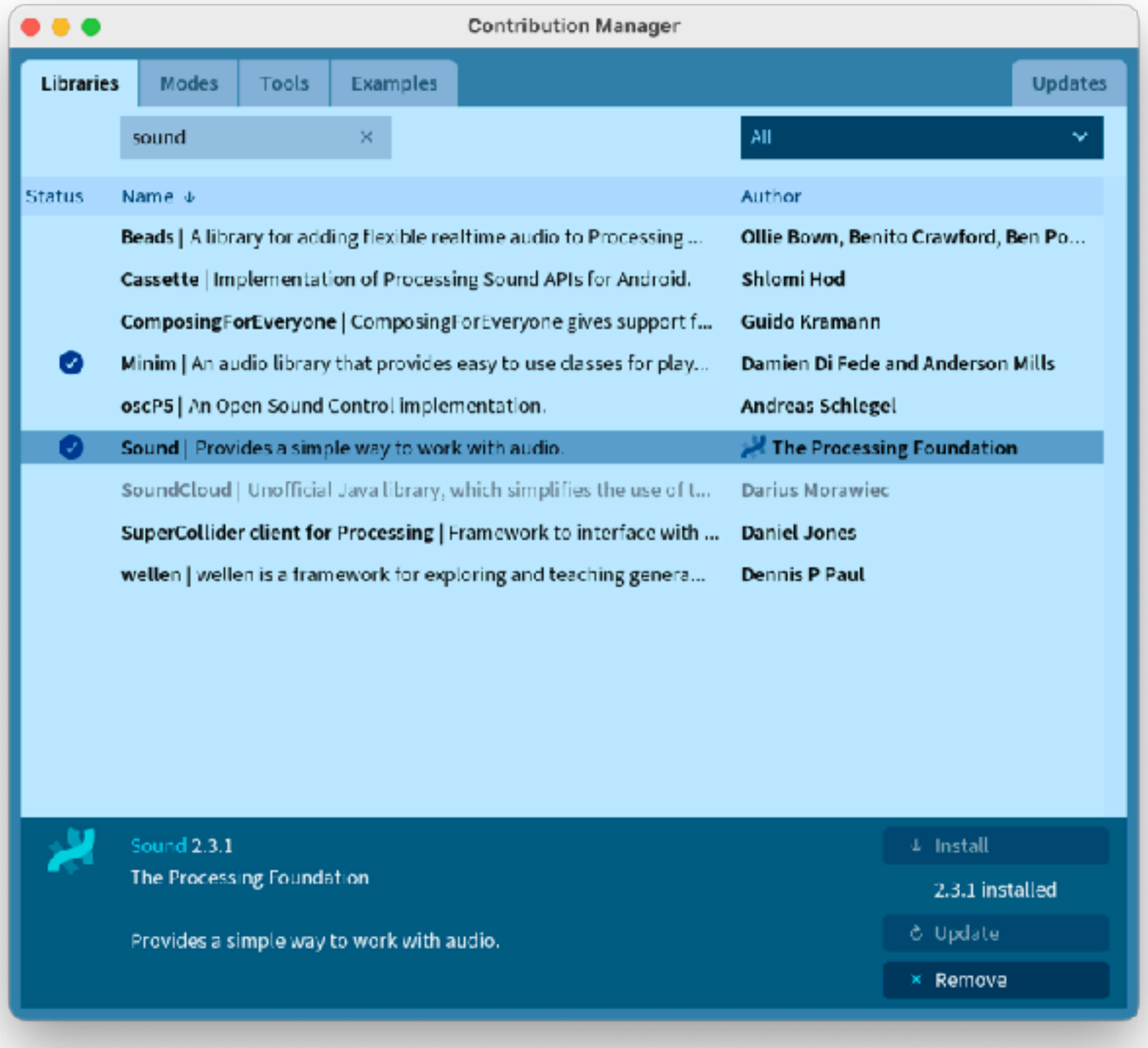
In order to work with sound information in Processing, we first need to **install an extension library** since Processing doesn't natively support this feature.

It can be simply installed by opening:

Sketch
→
Import Library
→
Add Library, typing 'sound' into the search field and then clicking on 'Install'.

`// Mac users`

`Sound library`



Contribution Manager

sound ×

All

Status	Name ↓	Author
	Beads | A library for adding flexible realtime audio to Processing ...	**Ollie Bown, Benito Crawford, Ben Po...**
	Cassette | Implementation of Processing Sound APIs for Android.	**Shlomi Hod**
	ComposingForEveryone | ComposingForEveryone gives support f...	**Guido Kramann**
✓	**Minim** | An audio library that provides easy to use classes for play...	**Damien Di Fede and Anderson Mills**
	oscP5 | An Open Sound Control implementation.	**Andreas Schlegel**
✓	**Sound** | Provides a simple way to work with audio.	**The Processing Foundation**
	SoundCloud | Unofficial Java library, which simplifies the use of t...	Darius Morawiec
	SuperCollider client for Processing | Framework to interface with ...	**Daniel Jones**
	wellen | wellen is a framework for exploring and teaching genera...	**Dennis P Paul**

Sound 2.3.1
The Processing Foundation

Provides a simple way to work with audio.

2.3.1 installed

You can find more in the document:

<u>`Processing Sound Library`</u>

Sound Reactive

Use this code before void setup:

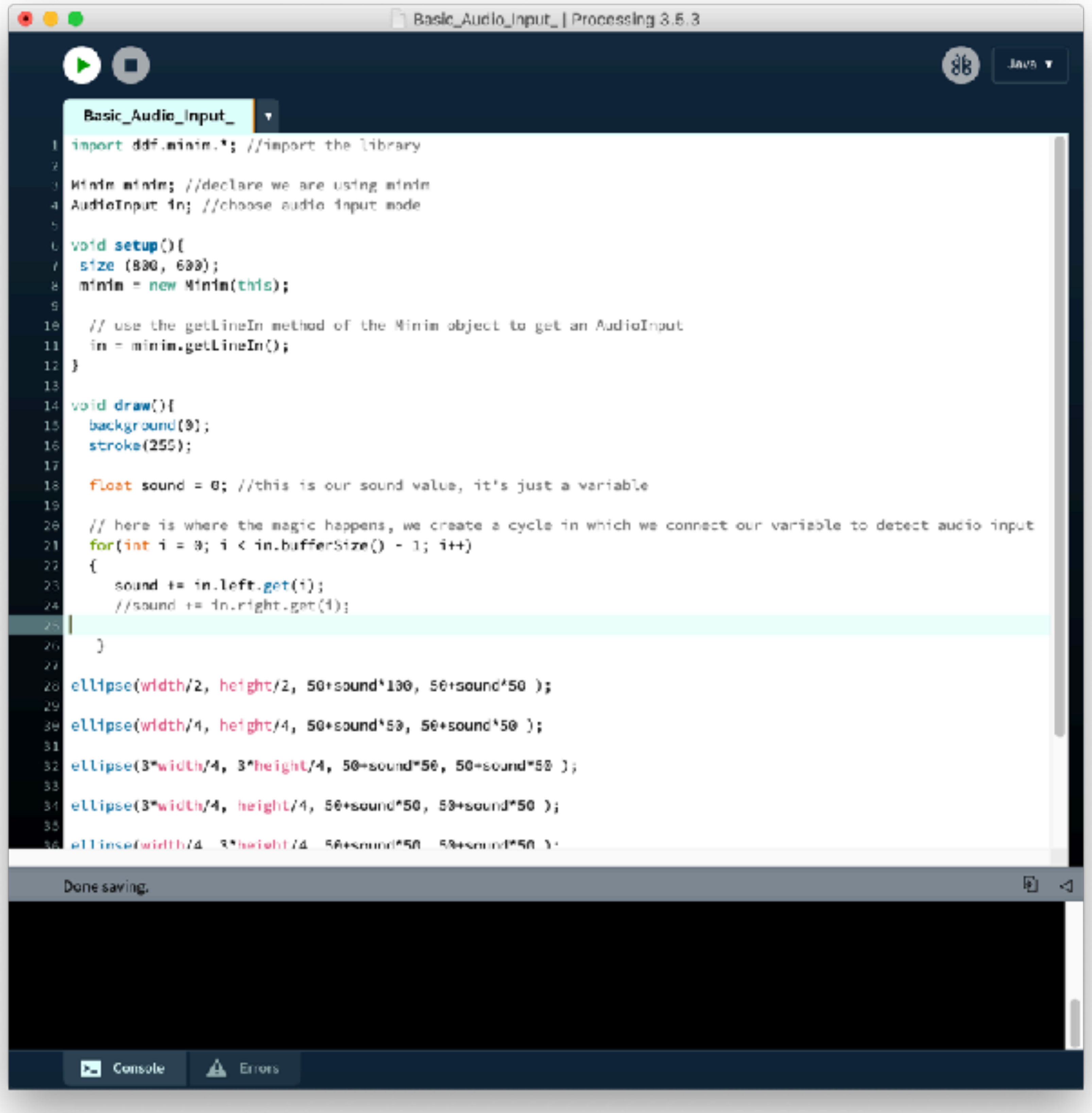
```
import ddf.minim.*; //import the library
Minim minim; //declare we are using minim
AudioInput in; //choose audio input mode
```

Use this code inside void setup:

```
minim = new Minim(this);
in = minim.getLineIn();
```

Use this code inside void draw:

```
float sound = 0;
 for(int i = 0; i < in.bufferSize() - 1; i++) {
   sound += in.left.get(i);
}
```



```
import ddf.minim.*; //import the library

Minim minim; //declare we are using minim
AudioInput in; //choose audio input mode

void setup(){
 size (800, 600);
 minim = new Minim(this);

  // use the getLineIn method of the Minim object to get an AudioInput
  in = minim.getLineIn();
}

void draw(){
  background(0);
  stroke(255);

  float sound = 0; //this is our sound value, it's just a variable

  // here is where the magic happens, we create a cycle in which we connect our variable to detect audio input
  for(int i = 0; i < in.bufferSize() - 1; i++)
  {
     sound += in.left.get(i);
     //sound += in.right.get(i);

   }

ellipse(width/2, height/2, 50+sound*100, 50+sound*50 );

ellipse(width/4, height/4, 50+sound*50, 50+sound*50 );

ellipse(3*width/4, 3*height/4, 50+sound*50, 50+sound*50 );

ellipse(3*width/4, height/4, 50+sound*50, 50+sound*50 );

ellipse(width/4, 3*height/4, 50+sound*50, 50+sound*50 );
```

Done saving.

Connect audio input

now you can connect the variable sound you just created to your shapes

Equalizer

```
// open the folder of examples

sketch_6BASIC_AUDIO_INPUT
```

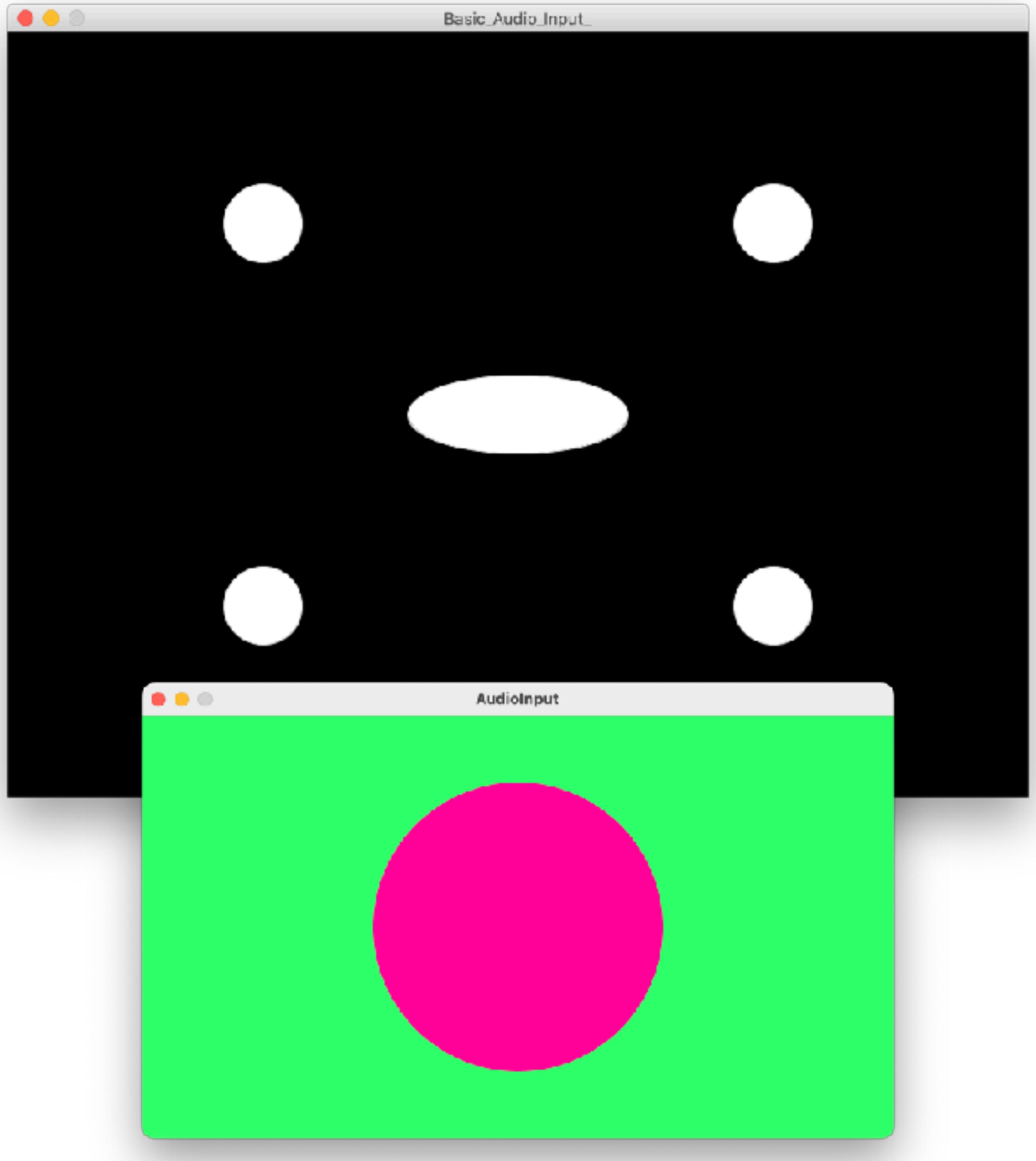
here you can find

Basic principles of sound reactivity

to understand how to import a library and how to implement sound reaction in shapes

tips:

try to have different reactions for different shapes, using math to adjust sensibility



Sound Reactive

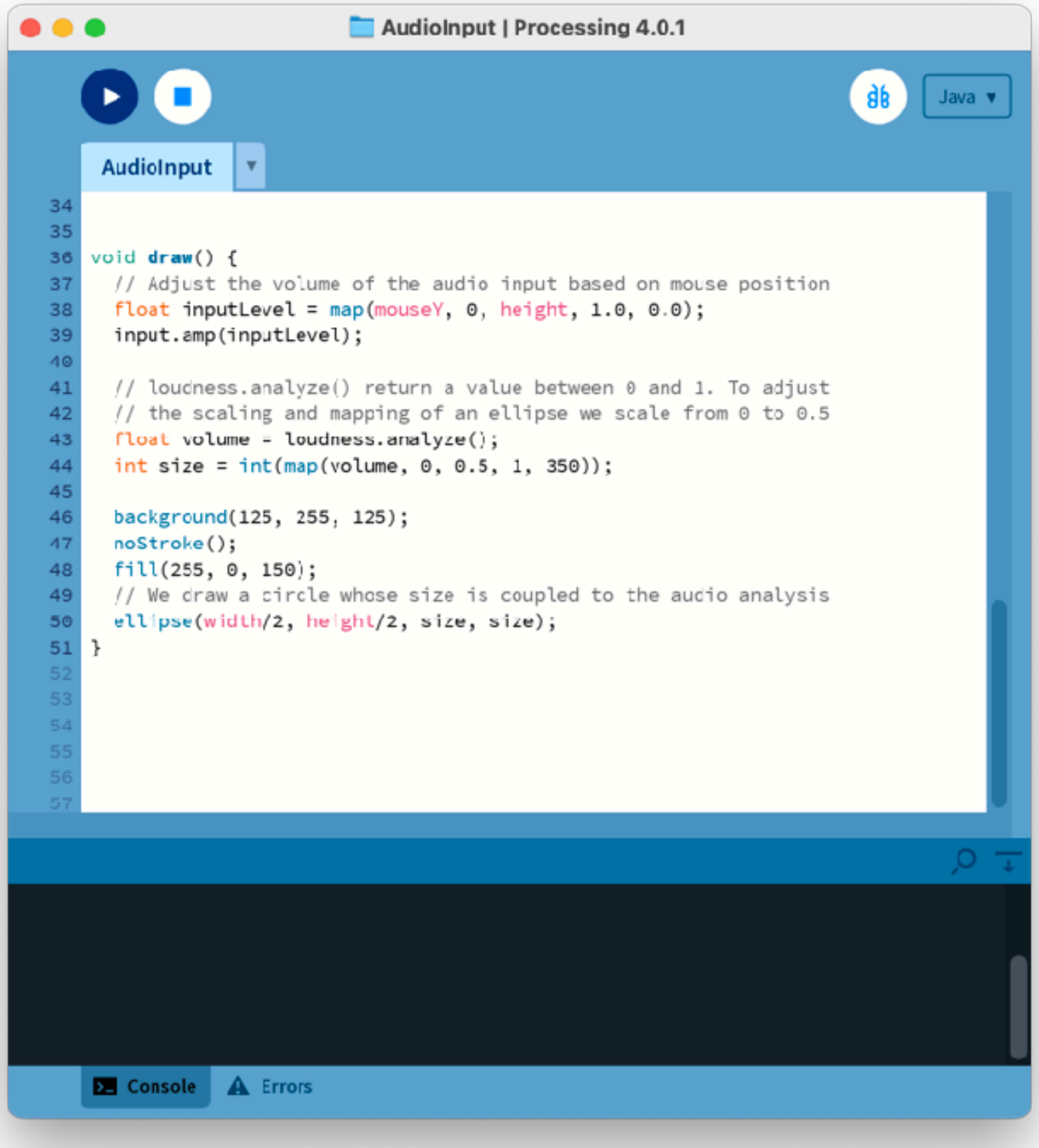
Go to Files - Examples

Libraries folder

Sound

IO

AudioInput



```
void draw() {
  // Adjust the volume of the audio input based on mouse position
  float inputLevel = map(mouseY, 0, height, 1.0, 0.0);
  input.amp(inputLevel);

  // loudness.analyze() return a value between 0 and 1. To adjust
  // the scaling and mapping of an ellipse we scale from 0 to 0.5
  float volume = loudness.analyze();
  int size = int(map(volume, 0, 0.5, 1, 350));

  background(125, 255, 125);
  noStroke();
  fill(255, 0, 150);
  // We draw a circle whose size is coupled to the audio analysis
  ellipse(width/2, height/2, size, size);
}
```

The logic is the same

The library is different
so the variables could be
different too

Be creative!

Let’s create your own!

here some sketches done by

student from past years

Here a link to a p5 game made by a student

https://editor.p5js.org/Cuchavira/sketches/diEpF0D7Z

`// open the folder past years' examples`

Credits: Luca Ghezzi, Andrea Zito, Hongni Ye

Thank You

giulio.interlandi@mail.polimi.it