

Département Informatique, CNAM
NSY116 - Multimédia et interaction humain-machine -2007-8

Introduction à Processing

P. Cubaud

1. L'environnement
2. Le langage
3. Les trois styles de programmation
4. Les libraries
5. Projets associés (sisters) : ex. avec Arduino



<http://processing.org>

<http://www.multimedialab.be/cours/logiciels/processing.htm>

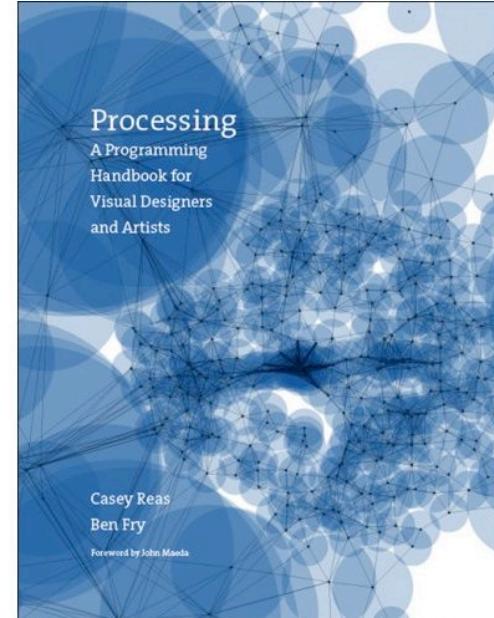


<http://reas.com/>

(ucla)

<http://benfry.com/>

(mit medialab)

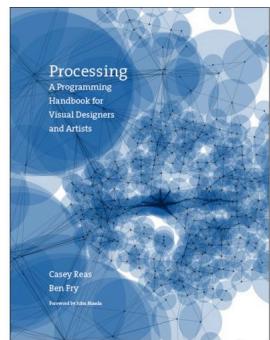


9/2007, 736 p.
MIT Press (35 euros)

We think most “integrated development environments” (Microsoft Visual Studio, Codewarrior, Eclipse, etc.) tend to be overkill for the type of audience we’re targeting with Processing. For this reason, we’ve introduced the ‘sketchbook’ which is a more lightweight way to organize projects. As trained designers, we’d like the process of coding to be a lot more like sketching. The sketchbook setup, and the idea of just sitting down and writing code (without having to write two pages to set up a graphics context, thread, etc) is a small step towards that goal. The idea of just writing a short piece of code that runs very easily (via a little run button) is a direct descendant of John Maeda’s work in [Design By Numbers](#), and our experiences maintaining it. (Yes, other languages and environments have done this first, but in our case, the concept is drawn from DBN).

<http://processing.org/faq.html>

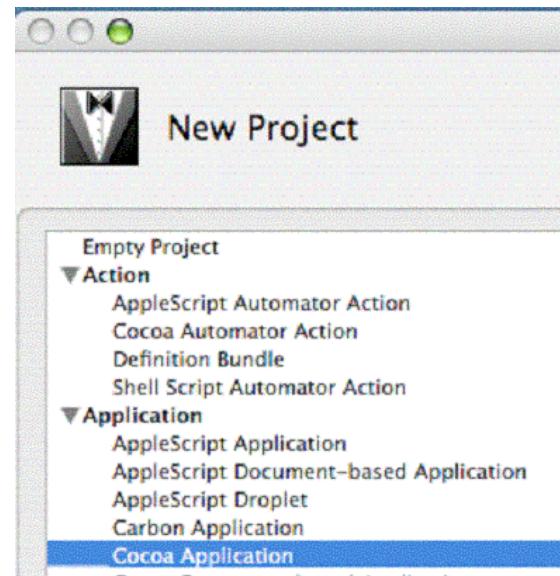
Sketching is necessary for the development of ideas. It is necessary to sketch in a medium related to the final medium so the sketch can approximate the outcome. Painters may construct elaborate drawings and sketches before executing the final work. Architects traditionally work first in cardboard and wood to better understand their forms in space. Musicians often work with a piano before scoring a more complex composition. To sketch electronic media, it's important to work with electronic materials. Just as each programming language is a distinct material, some are better for sketching than others, and artists working in software need environments for working through their ideas before writing final code. Processing is built to act as a software sketchbook, making it easy to explore and refine many different ideas within a short period of time.



(p. 2)

Peut-on aller au-delà du « sketch » ?

Digression sur Xcode (et autres IDE)

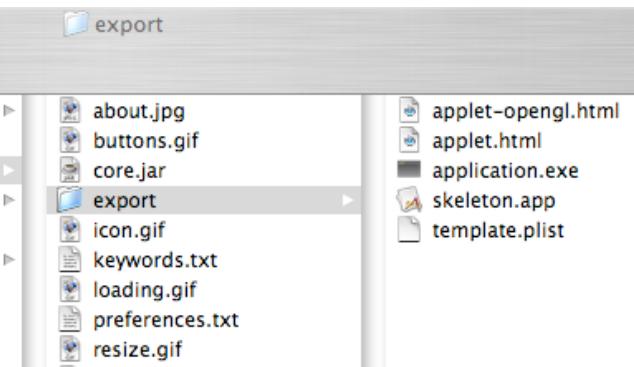
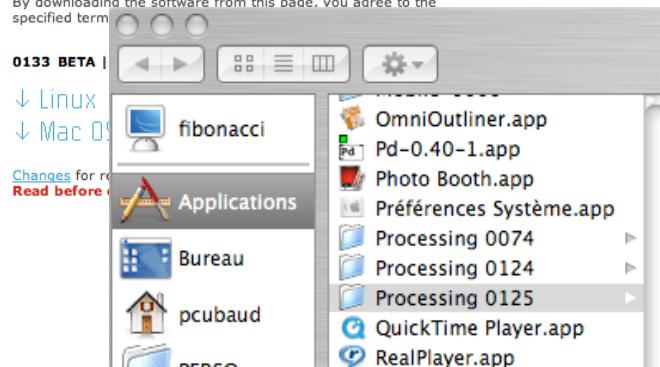


Digression sur Flash

A screenshot of the Adobe website's 'Comment acheter' (How to Buy) page. The top navigation bar includes links for 'SOLUTIONS', 'PRODUITS', 'SUPPORT', 'COMMUNAUTÉS', 'SOCIÉTÉ', 'TÉLÉCHARGER', 'STORE', and a search bar. The main content area has three sections: 'Adobe Stores' (with text 'Achetez en ligne via Adobe Store dans votre pays'), 'Programme d'achat de licences' (with text 'Réduisez vos dépenses en achetant les produits Adobe en volume grâce au programme d'achat de licences Adobe® Open Options.'), and 'Comment acheter' (with text 'Découvrez comment acheter les produits Adobe dans votre pays' and a dropdown menu set to 'France' with a 'Go' button). A footer link 'Rechercher un revendeur près de chez' is also visible.

Installation (très simple)

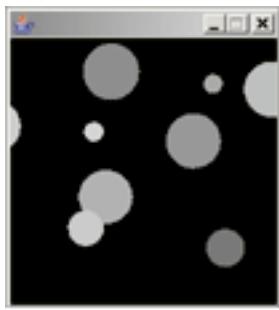
Platforms. The Processing Environment (IDE) runs on various Linux, Windows, and Mac OS X operating systems. Programs written with Processing run on various versions of Java.



REV 0068 - 2 février 2004
REV 0133 - 26 octobre 2007

Et toujours en version Beta pour la 1.0 !!

Processing Development Environment (PDE)



Display Window

Menu
Toolbar
Tabs

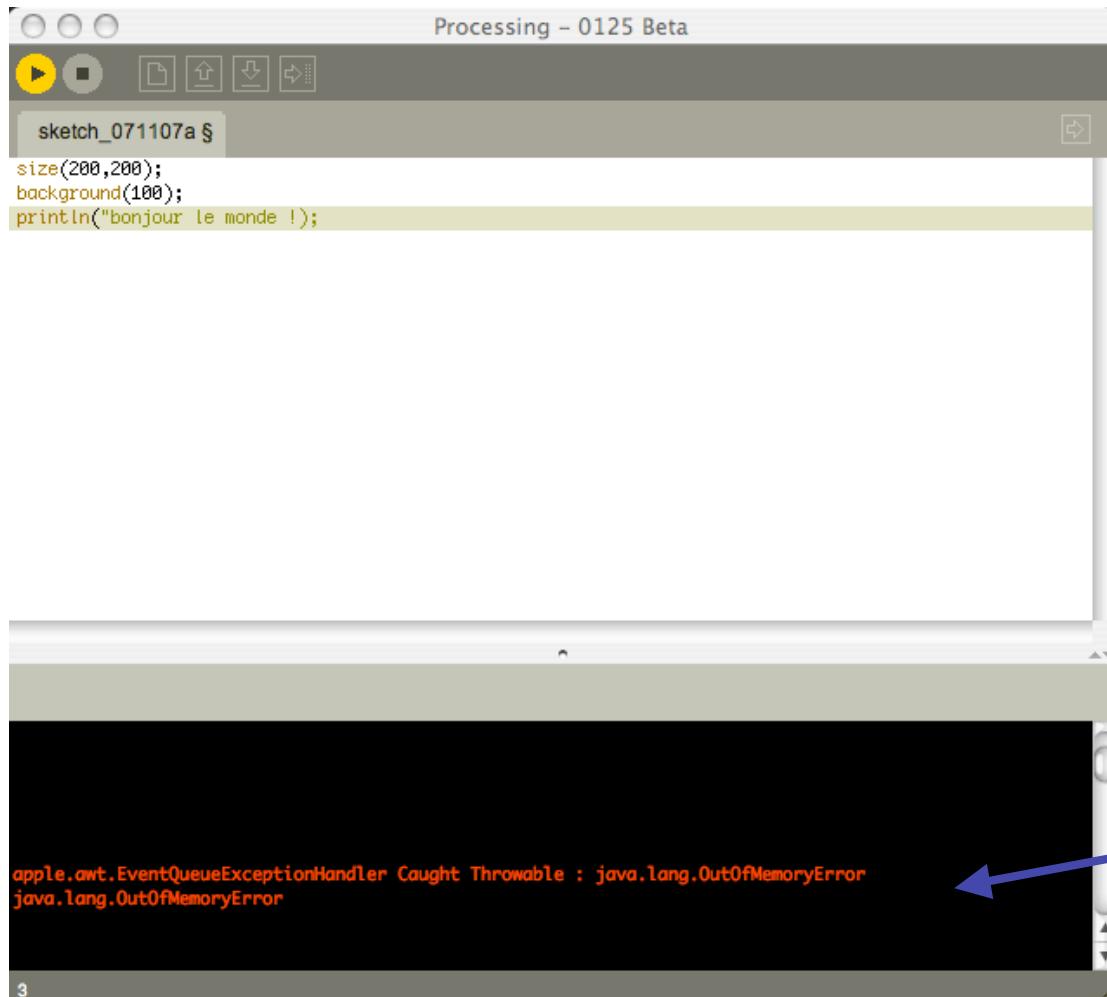
Text Editor

Message Area

Text Area

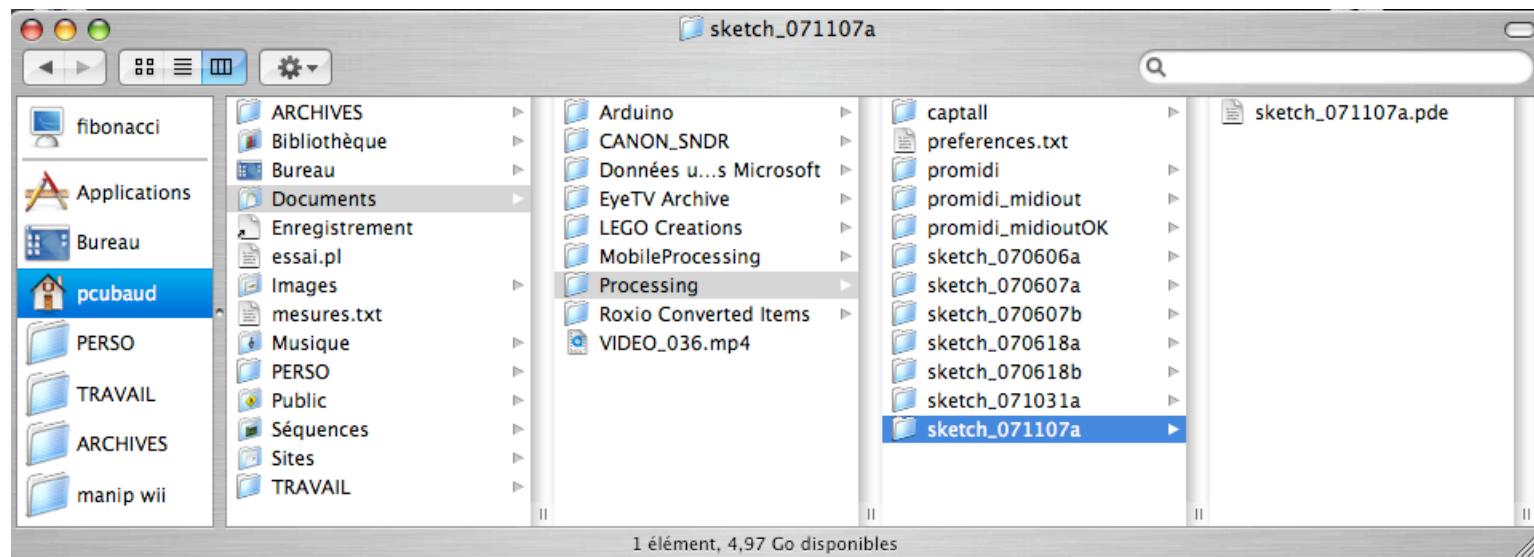
<http://processing.org/reference/environment/index.html>

Bonjour Monde

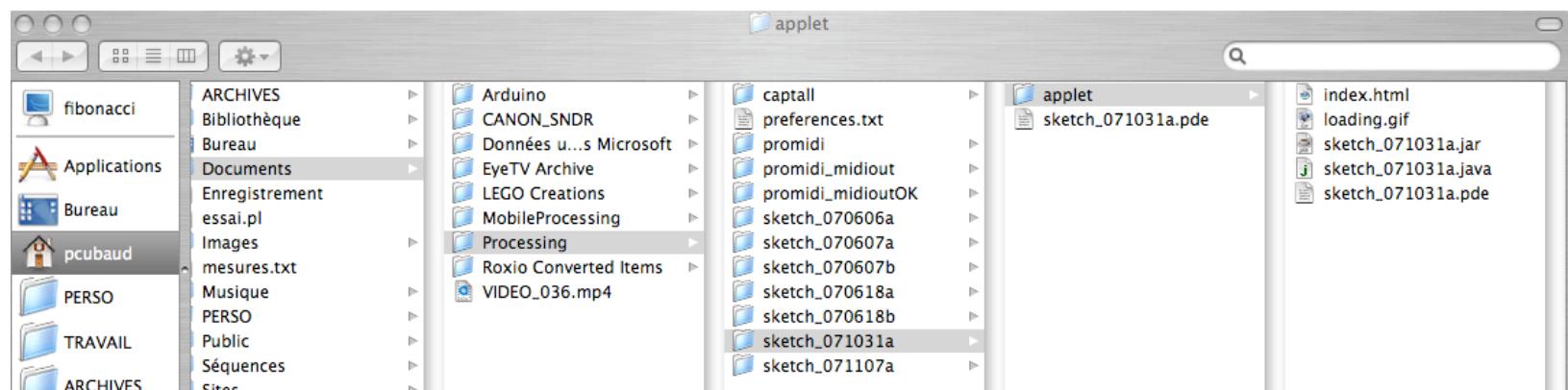


hum...

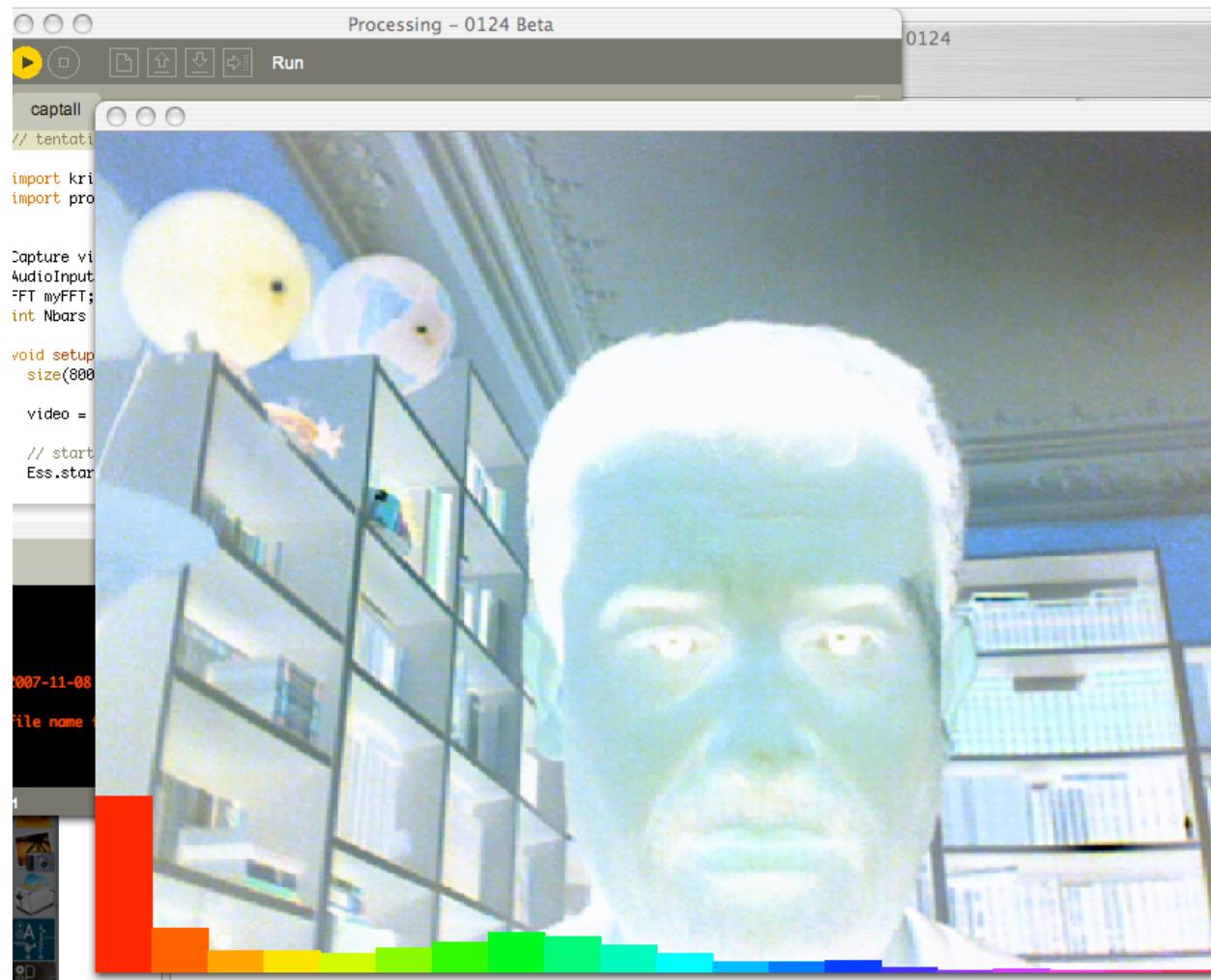
Après la sauvegarde : on crée un fichier .pde rangé par défaut dans le dossier « Documents »



Après un export : création d'un répertoire « applet » :



Bonjour Monde, le retour



Code très court !

```
// tentative de modulateur PC juin 2007

import krister.Ess.*;
import processing.video.*;

Capture video;
AudioInput myInput;
FFT myFFT;
int Nbars = 20;

void setup() {
    size(800,600);
    noStroke();
    Ess.start(this);
    myFFT=new FFT(512);
    myFFT.limits();
    myFFT.averages(Nbars);
    myInput=new AudioInput(512);
    myInput.start();
}
```

Vidéo ?
Son ?
Dessin ?

```
void draw() {
    background(0,0,Nbars);
    video.read();
    myFFT.getSpectrum(myInput);
    // coloriage de la video en fct de la fft
    tint(10000*myFFT.maxAverages[0],
        10000*myFFT.maxAverages[1],
        10000*myFFT.maxAverages[2]);
    image(video, 0, 0);
    colorMode(HSB,Nbars);
    for (int i=0; i<Nbars; i++) {
        float x = i*width/Nbars;
        //float y = height*(1.0-myFFT.maxAverages[i]);
        float w = width/Nbars;
        float h = height*myFFT.maxAverages[i];
        fill(i,Nbars,Nbars);
        rect(x,height-h,w+0.5,h);
    }
    colorMode(RGB, 255);
}

public void stop() {
    Ess.stop();
    super.stop();
}

public void audioInputData(AudioInput theInput) {
    myFFT.getSpectrum(myInput);
}
```

2. Le langage (par l'exemple...)

Manuel de ref.

dans le livre



Language (API) \ Processing 1.0 (BETA)
file:///Applications/Processing%200124/reference/index.html

API \ Processing...

keyPressed

Attributes

Time & Date

Math

Operators

Text Area

Image

Calculation

Control

Conditional Operators

Action

Conditionals

Logical Operators

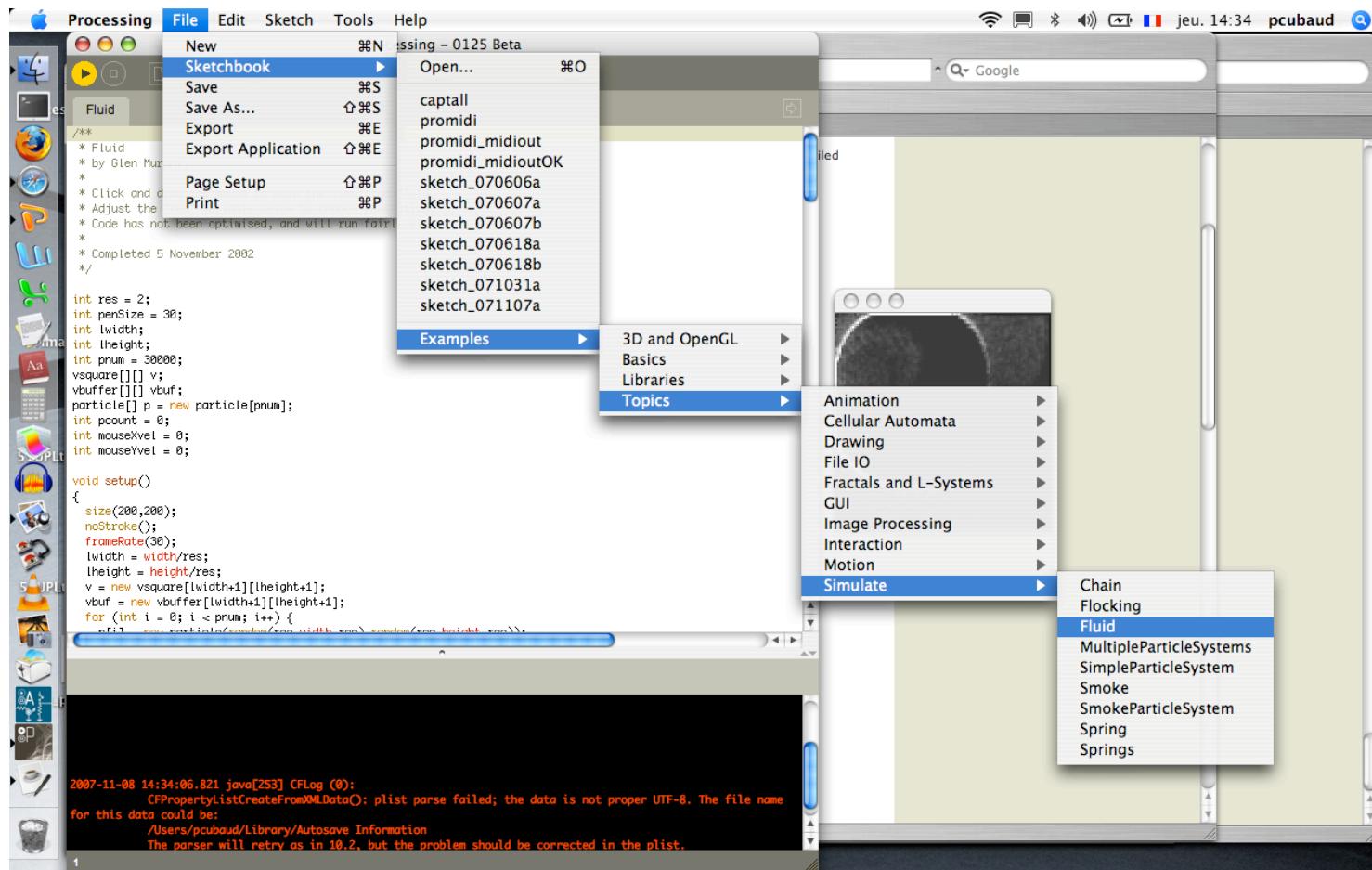
Transform

Trigonometry

Random

Routeur **webmailCNAM** **googleBooks** **AbeBooks** **Amazon** **Apple** **eBay** **anteres** **UTILES**

Un exemple de sketch dans le dossier *examples*, parmi 225 (*)



(*) cd /Applications/Processing\ 0.125/examples/ ;
ls -lR | grep pde | wc -l

Processing par rapport à Java

Input

```
mouseX  
mouseY
```

```
void mousePressed() {  
    // Statements  
}
```

```
if (key == 'a') {  
    // Statements  
}
```

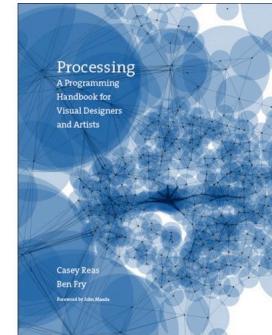
/* Assuming there are two variables in the program named mouseX and mouseY, these values must be changed by the programmer in the mouseMoved() and mouseDragged methods. */

```
public void mouseMoved(MouseEvent e) {  
    mouseX = e.getX();  
    mouseY = e.getY();  
}
```

```
public void mouseDragged(MouseEvent e) {  
    mouseX = e.getX();  
    mouseY = e.getY();  
}
```

```
public void mousePressed(MouseEvent e) {  
    // Statements  
}
```

```
public void keyPressed(KeyEvent e) {  
    char key = e.getKeyChar();  
    if (key == 'a') {  
        // Statements  
    }  
}
```



p. 688

Processing par rapport à Python

```
// calcul des coef. transformee de Fourier

float[] u = {0,0.262,0.524,0.786,1.047,1.309,0,-1.309,-1.047,-0.786,-0.524,-0.262};

int N = 12;
int R = 6;
float a = 0;          ##### calcul des coef. transformee de Fourier

for (int k=0;k<N;k++) {
    a += u[k];
}
a *= 1.0/N;
println("a0 : "+a);      u = [0,0.262,0.524,0.786,1.047,1.309,0,-1.309,-1.047,-0.786,-0.524,-0.262]
                           N = 12
                           R = 6
                           a = 0
                           for k in range(0,N) :
                               a += u[k]
                               a *= 1/N
                               print "a0 : ",a
```

Pas juste une affaire de crochets et d'accolades !

Types élémentaires :

Name Size Value range

boolean 1 bit true or false

byte 8 bits -128 to 127

char 16 bits 0 to 65535

int 32 bits -2,147,483,648 to 2,147,483,647

float 32 bits 3.40282347E+38 to -3.40282347E+38

color 32 bits 16,777,216 colors

Types constructeurs : String, array, objets

Contrôle du flot : if-else, while, for ...

Fonctions prédéfinies usuelles (chaines, math, temps, fichiers ...)

Fonctions (avec surcharge, récurrence possible)

3. Les trois styles de programmation

1) Basic

This mode is used drawing static images and learning fundamentals of programming. Simple lines of code have a direct representation on the screen.

2) Continuous

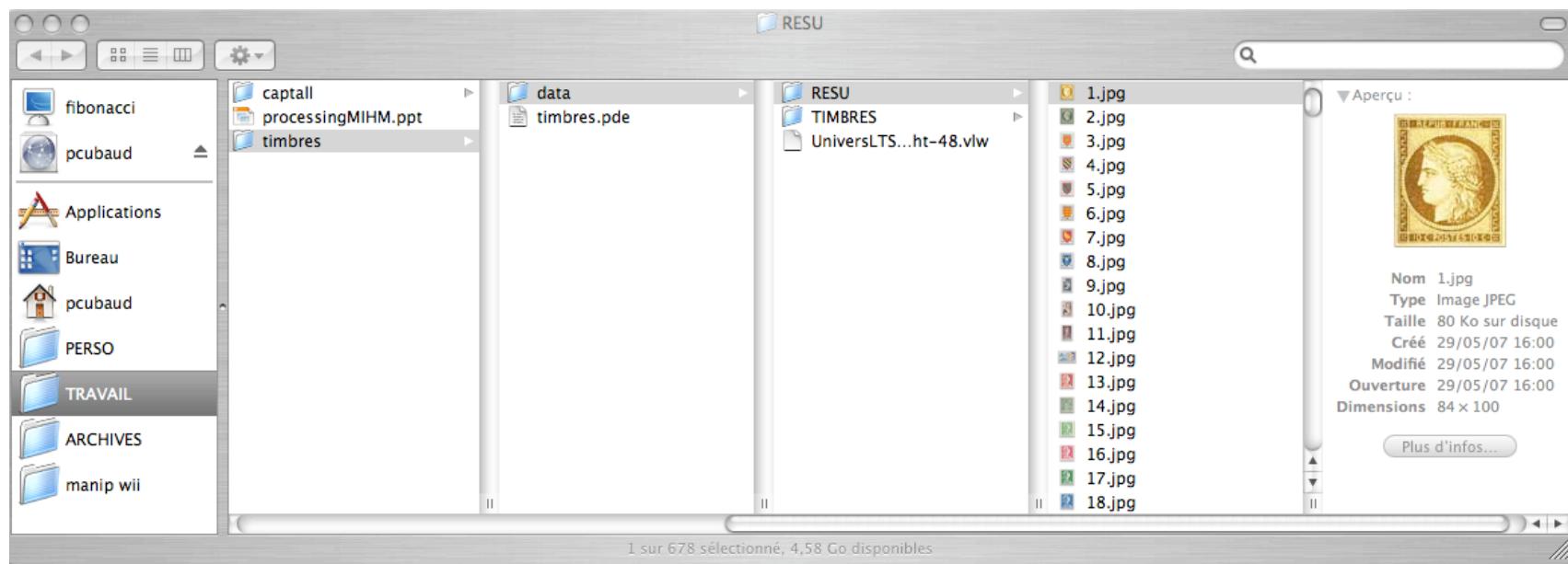
This mode provides a `setup()` structure that is run once when the program begins and a `draw()` structure which by default continually loops through the code inside. This additional structure allows writing custom functions and classes and using keyboard and mouse events.

3) Java

This mode is the most flexible, allowing complete Java programs to be written from inside the Processing Environment. Writing in Java Mode removes the limitations of the Processing Libraries and gives access to the full Java programming language.

3.1 Exemple en mode basic : fabriquer des planches de timbres

motivation : travaux de recherche au CNAM/CEDRIC sur la visualisation de grandes masses de documents, pour en particulier encourager la découverte accidentelle



Ici, un répertoire avec 678 imagettes de timbres

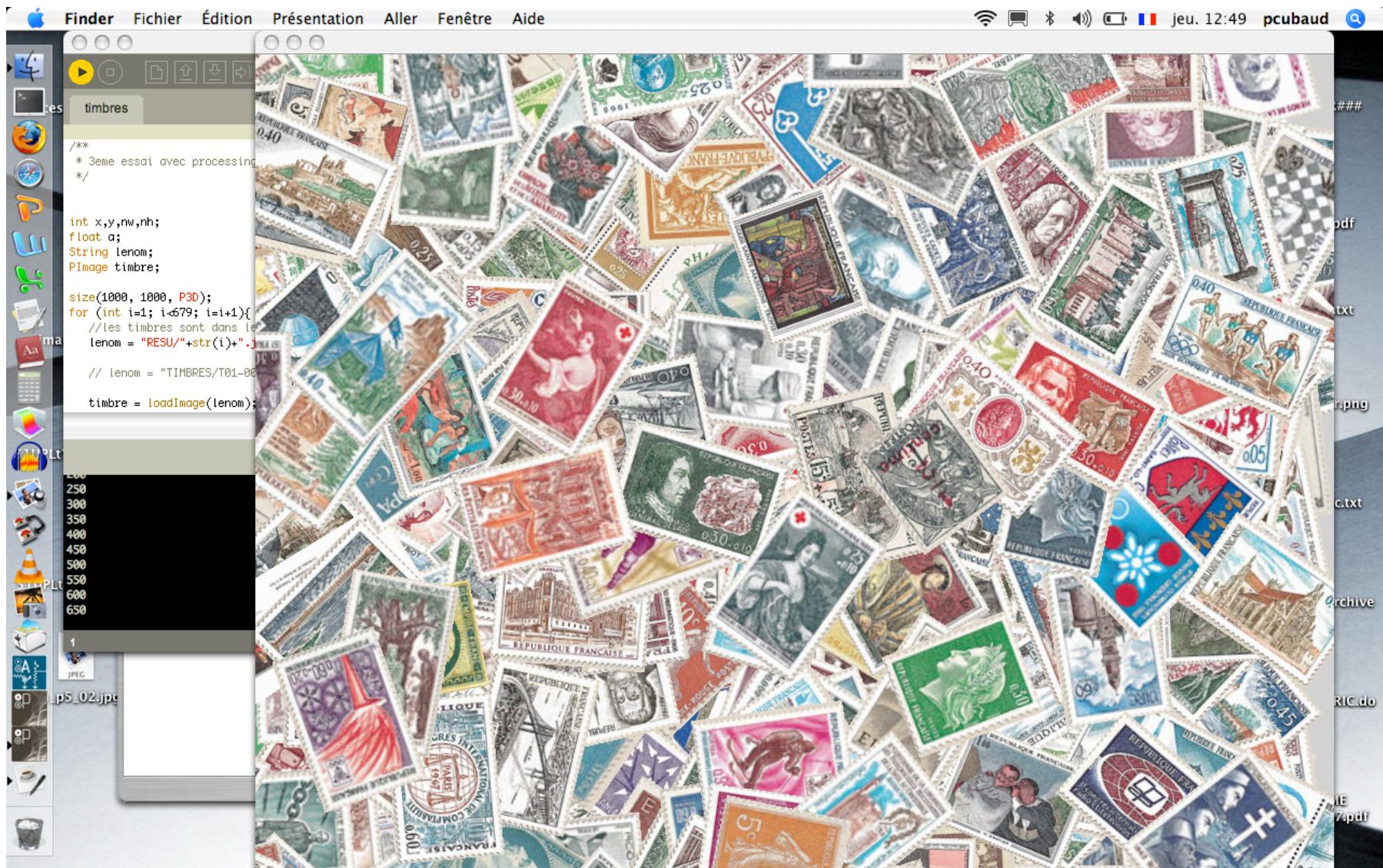
Le code

(commenter)

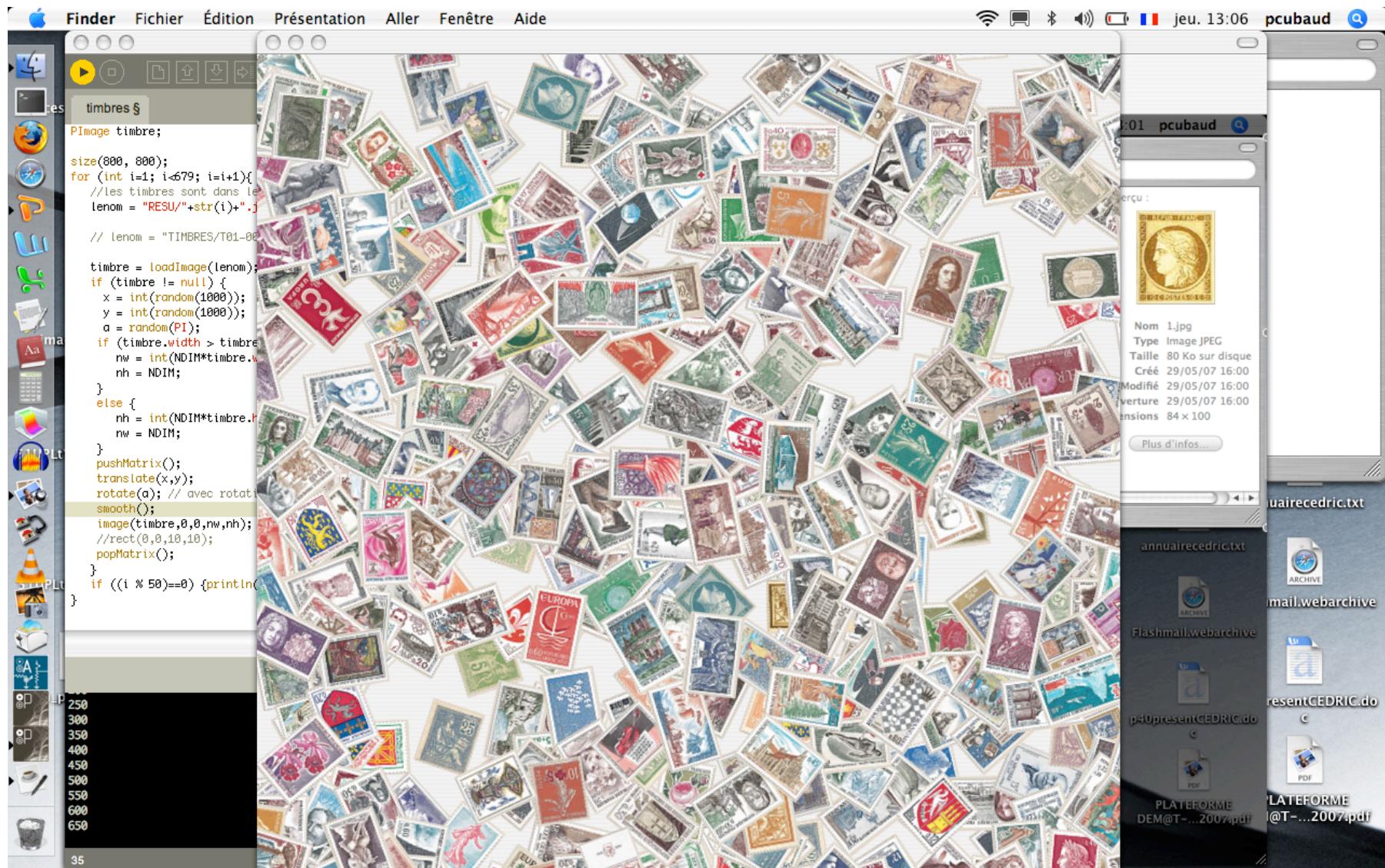
```
Int NDIM = 100;
int x,y,nw,nh;
float a;
String lenom;
PImage timbre;

size(1000, 1000, P3D);
for (int i=1; i<679; i=i+1){
    //les timbres sont dans le repertoire RESU
    lenom = "RESU/" + str(i) + ".jpg";
    timbre = loadImage(lenom);
    if (timbre != null) {
        x = int(random(1000));
        y = int(random(1000));
        a = random(PI);
        if (timbre.width > timbre.height) {
            nw = int(NDIM*timbre.width/timbre.height);
            nh = NDIM;
        }
        else {
            nh = int(NDIM*timbre.height/timbre.width);
            nw = NDIM;
        }
        pushMatrix();
        translate(x,y);
        rotate(a); // avec rotation, sinon mettre en commentaire
        //smooth();
        image(timbre,0,0,nw,nh);
        popMatrix();
    }
    if ((i % 50)==0) {println(i);}
}
```

Résultat (NDIM=100)

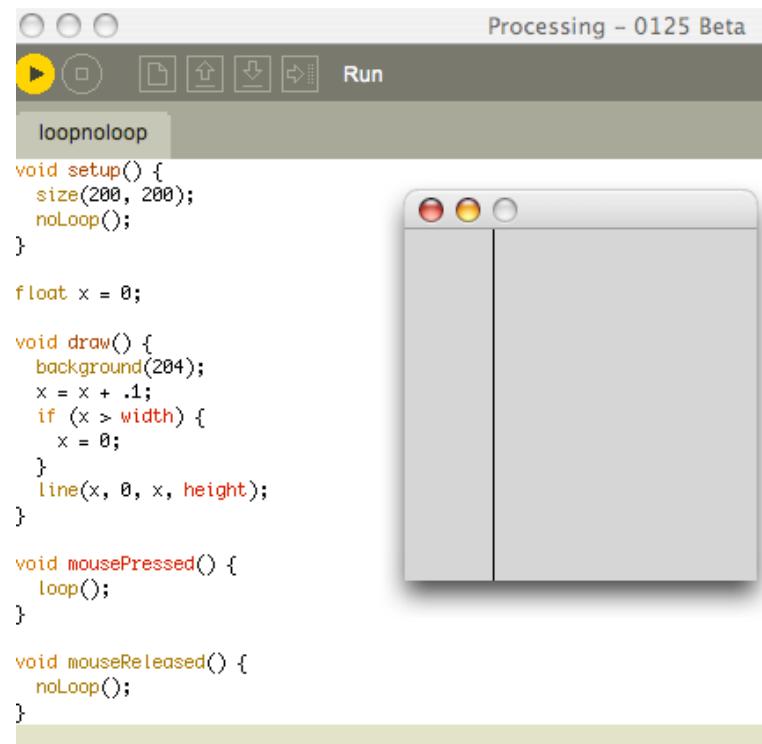


Résultat (NDIM=50)



3.2 Exemple en mode continuous (extrait du manuel de ref.)

```
void setup() {  
    size(200, 200);  
    noLoop();  
}  
  
float x = 0;  
  
void draw() {  
    background(204);  
    x = x + .1;  
    if (x > width) {  
        x = 0;  
    }  
    line(x, 0, x, height);  
}  
  
void mousePressed() {  
    loop();  
}  
  
void mouseReleased() {  
    noLoop();  
}
```



Un autre exemple avec gestion du temps

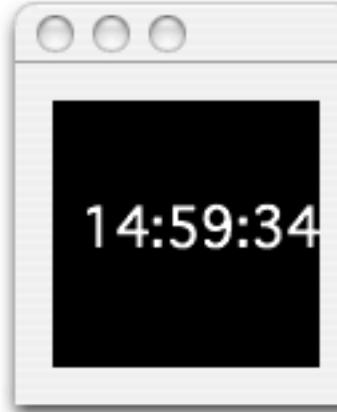
Processing - 0125 Beta

Play

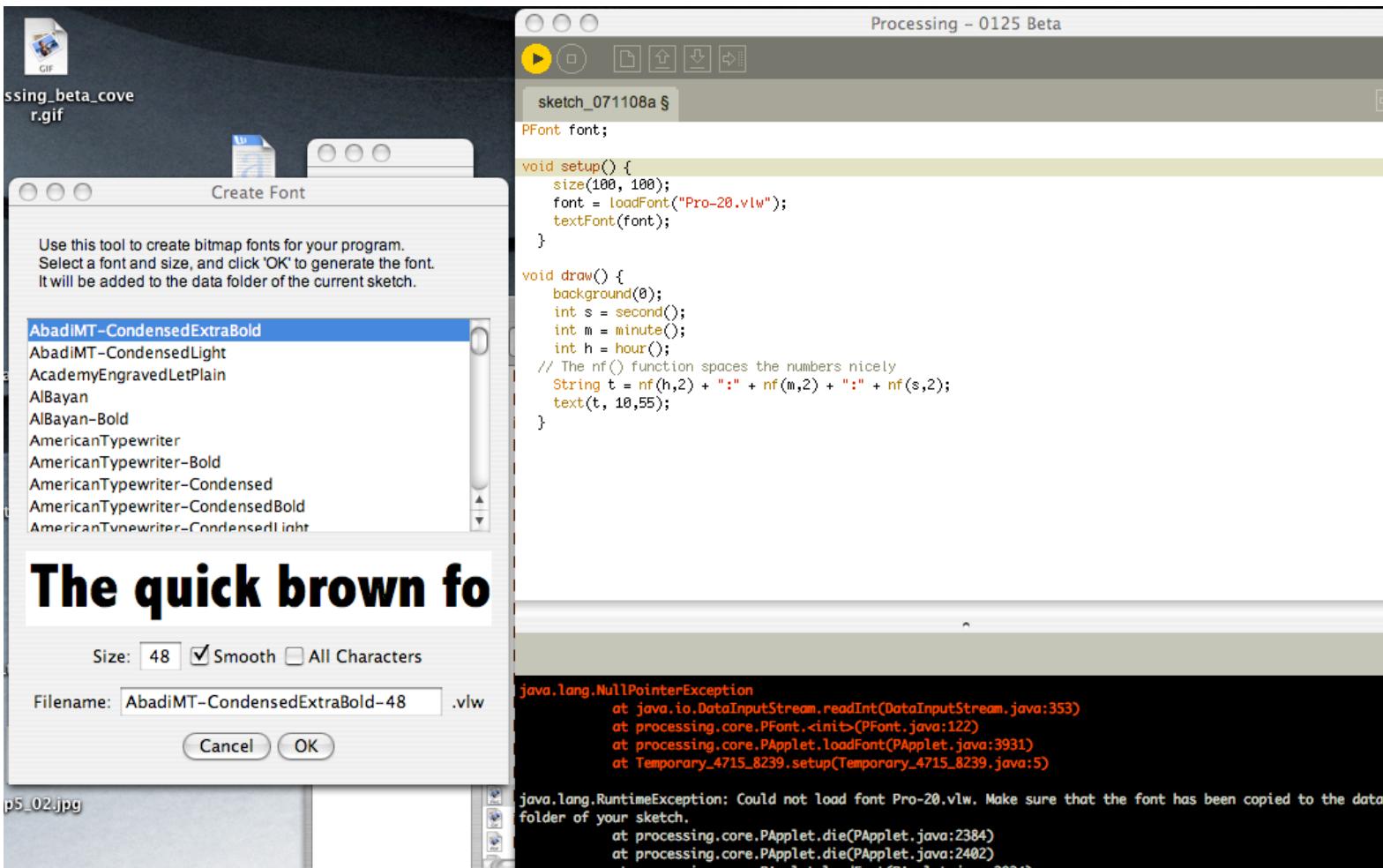
File Editor §

```
PFont font;
```

```
void setup() {
    size(100, 100);
    font = loadFont("Geneva-20.vlw");
    textAlign(CENTER);
}
void draw() {
    background(0);
    int s = second();
    int m = minute();
    int h = hour();
    // The nf() function spaces the numbers nicely
    String t = nf(h,2) + ":" + nf(m,2) + ":" + nf(s,2);
    text(t, 10,55);
}
```

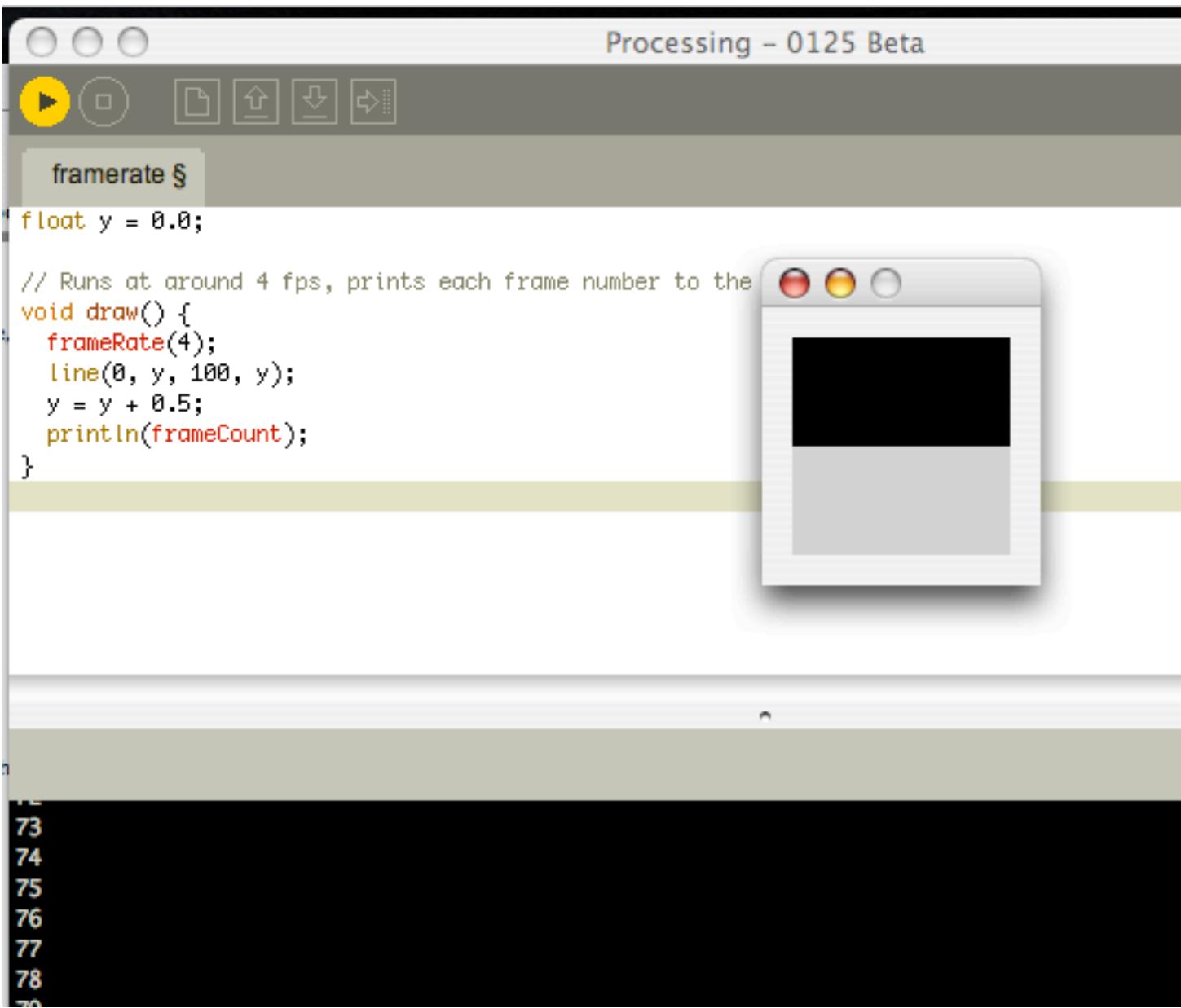


Digression sur les fontes



Générées par un outil de l'env.
Ajoutées dans un dossier *data* du sketch

Digression sur le frame-rate



The screenshot shows the Processing 0125 Beta IDE interface. The title bar reads "Processing - 0125 Beta". Below the title bar is a toolbar with various icons: play, stop, step, zoom, and file operations. A text input field contains the text "framerate §". The main code editor window displays the following code:

```
float y = 0.0;

// Runs at around 4 fps, prints each frame number to the
void draw() {
    frameRate(4);
    line(0, y, 100, y);
    y = y + 0.5;
    println(frameCount);
}
```

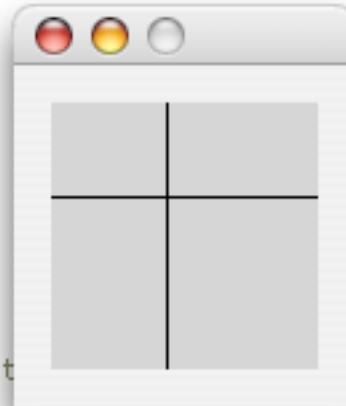
To the right of the code editor is a preview window showing a black rectangle at the top and a gray rectangle below it. The bottom portion of the preview window is dark, indicating it is not currently active. The status bar at the bottom of the IDE shows the line numbers 73 through 79.

un autre exemple

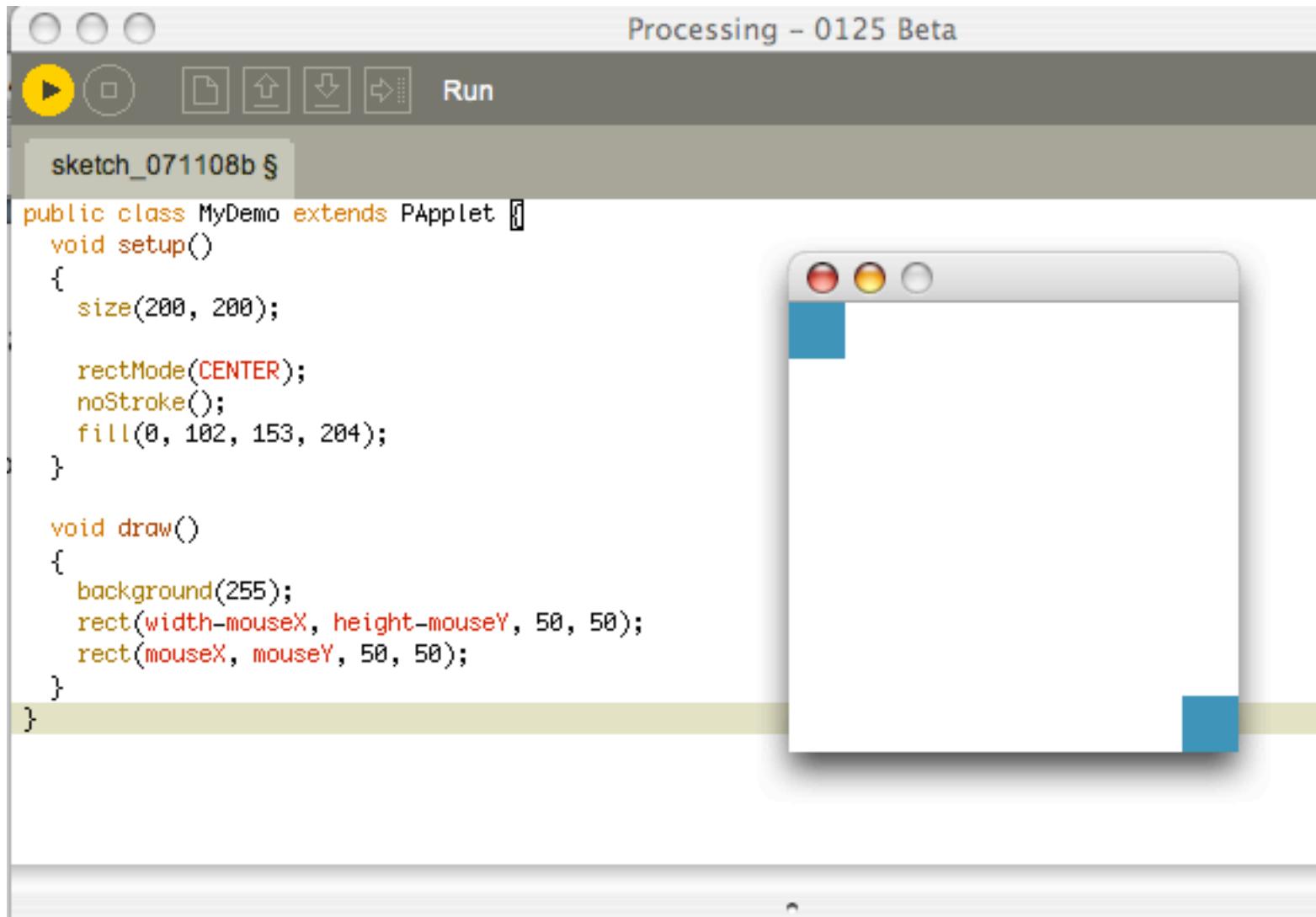
Processing – 0125 Beta

sketch_071108b §

```
int frame = 0;
void setup() {
    size(100, 100);
    frameRate(30);
}
void draw() {
    if (frame > 60) { // If 60 frames since the mouse
        noLoop(); // was pressed, stop the program
        background(0); // and turn the background black.
    } else { // Otherwise, set the background
        background(204); // to light gray and draw lines
        line(mouseX, 0, mouseX, width); // at the mouse position
        line(0, mouseY, height, mouseY);
        frame++;
    }
}
void mousePressed() {
    loop();
    frame = 0;
}
```



3.3 Exemple en mode java (manuel de ref.)



The screenshot shows the Processing IDE interface. The title bar reads "Processing - 0125 Beta". Below the title bar is a toolbar with icons for play, stop, file operations, and run. The central area displays a Java code snippet for a PApplet:

```
public class MyDemo extends PApplet {
    void setup()
    {
        size(200, 200);

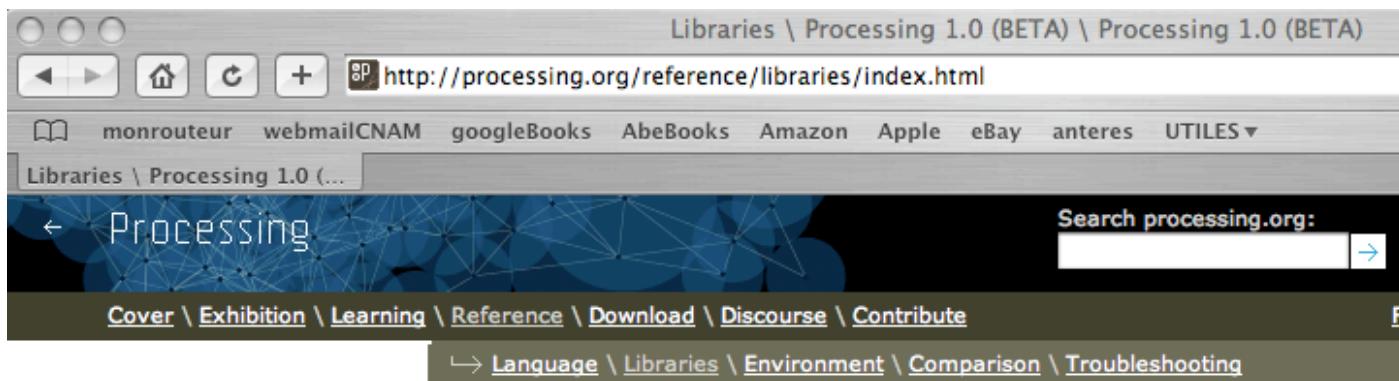
        rectMode(CENTER);
        noStroke();
        fill(0, 102, 153, 204);
    }

    void draw()
    {
        background(255);
        rect(width-mouseX, height-mouseY, 50, 50);
        rect(mouseX, mouseY, 50, 50);
    }
}
```

To the right of the code editor is a preview window showing a white canvas with two small blue rectangles. One rectangle is centered at the top-left of the canvas, and the other is centered at the bottom-right.

On peut utiliser un autre IDE, comme Eclipse

4. Les librairies : de base (core) ou tierces (contributors)



Libraries. Extending Processing beyond graphics and image, libraries enable audio, video, and communicating with other devices.

Core Libraries

[Video](#)

Interface to Apple's QuickTime for using a camera, playing movie files, and creating movies.

[Network](#)

Sending and receiving data via the Internet through the creation of simple clients and servers.

[Serial](#)

Supports sending data between Processing and external hardware via serial communication (RS-232).

[Candy SVG Import](#)

A minimal SVG file importer. Loads and displays vector files.

[XML Import](#)

A minimal XML importer.

[OpenGL](#)

Support for exporting OpenGL accelerated sketches. Utilizes the JOGL library.

[PDF Export](#)

Generates PDF files.

[DXF Export](#)

Lines and triangles from P3D or OPENGL rendering modes can be sent directly to a DXF file.

[» Netscape.JavaScript](#)

Methods for interfacing between Javascript and Java Applets exported from Processing.

Exemple de librairie de base : 3D sur opengl

Processing - 0125 Beta

TexturedSphere

```
/*
 * Textured Sphere
 * by Mike 'Flux' Chang (cleaned up by Aaron Koblin).
 * Based on code by Toxi.
 *
 * A 3D textured sphere with simple rotation control.
 * Note: Controls will be inverted when sphere is upside down.
 * Use an "arc ball" to deal with this appropriately.
 */

import processing.opengl.*;

PImage bg;
PImage texmap;

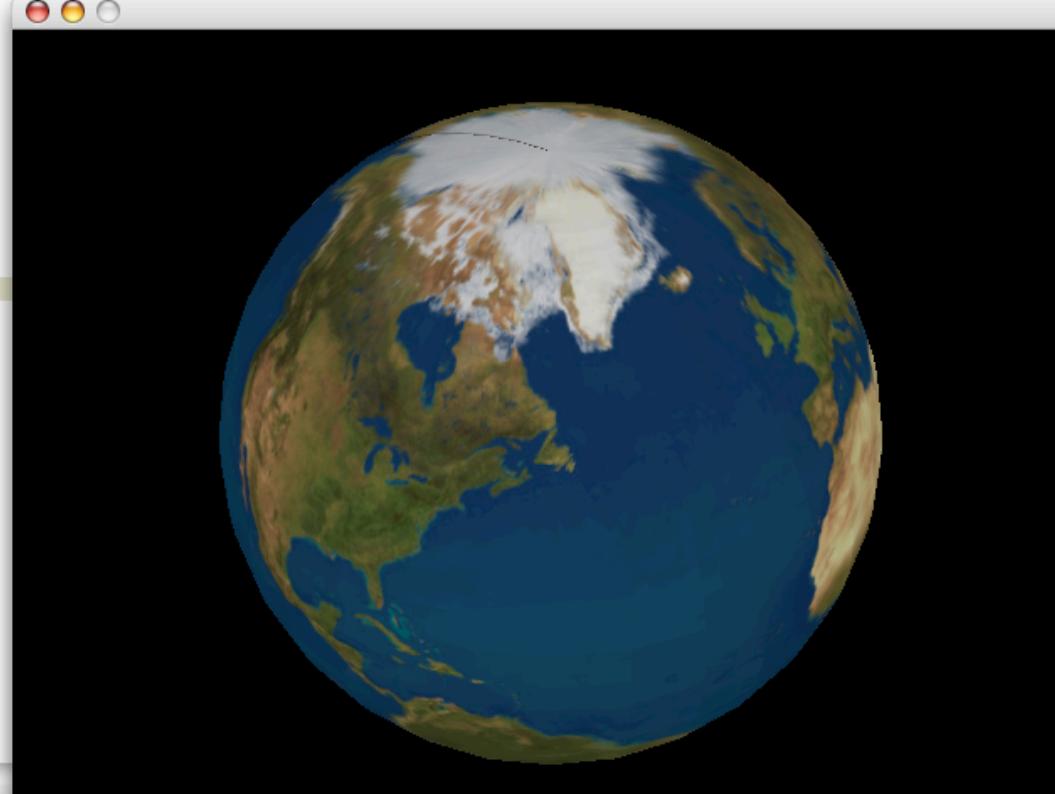
int sDetail = 35; //Sphere detail setting
float rotationX = 0;
float rotationY = 0;
float velocityX = 0;
float velocityY = 0;
float globeRadius = 300;
float pushBack = 0;

float[] cx,cz,sphereX,sphereY,sphereZ;
float sinLUT[];
float cosLUT[];
float SINCOS_PRECISION = 0.5f;
int SINCOS_LENGTH = int(360.0 / SINCOS_PRECISION);

void setup()
{
    size(640, 480, OPENGL);
    texmap = loadImage("world32k.jpg");
    initializeSphere(sDetail);
}

void draw()
{
    background(bg);
    sphere(cx,cz,sphereX,sphereY,sphereZ, sDetail);
}
```

Google



A 3D textured sphere visualization in Processing. The sphere is rendered with a detailed world map texture, showing continents and oceans. It is centered against a black background. The Processing IDE window shows the code for creating this visualization, including imports for OpenGL, variable declarations, and setup/draw functions. The draw function contains a call to the sphere() method with parameters for center, radius, and detail level.

2007-11-08 16:21:59.754 java[387] CFLog (0):
 CFPropertyListCreateFromXMLData()

Clist parse failed: the data is not proper UTF-8. The file name

Les contribs : plus d'une cinquantaine à ce jour !

Assez inégal

Contributions

Sound

» [Minim](#)

by [Damien Di Fede](#)

Uses the JavaSound API to provide an easy-to-use audio library. A simple API while still providing a reasonable amount of flexibility for more advanced users.

» [Ess](#)

by [Krister Olsson](#)

Sound library that allows sound sample data to be loaded or streamed, generated in real-time, manipulated, saved, analyzed or simply played back.

» [jm-Etude](#)

by Daniel Dihardja

Provides functions to communicate with [jMusic](#) for easier music composition programming.

» [Sonia](#)

by [Amit Pitaru](#)

Audio library for sound playback and synthesis. Integrates [Jsyn](#) and requires a browser plugin for playback.

» [Sonia Helper](#)

by [Marius Watz](#)

Normalizes and damps the FFT analysis values produced by Sonia. This is built into Ess.

Computer Vision

» [BlobDetection](#)

by [v3ga](#)

Performs the computer vision technique of finding "blobs" in an image.

» [reactIVision TUIO](#)

by [Martin Kaltenbrunner](#)

Interface to the reactIVision vision engine for object tracking.

» [LibCV](#)

by [toxi](#)

Grabs video frames from a camera using the Java Media Framework (JMF). Does not require QuickTime or WinVDIG for Windows machines.

» [JMyron \(WebCamXtra\)](#)

by [Josh Nimoy](#) et al.

Camera library for motion detection, color tracking, blob distinction, and pixel addressing. Does not require QuickTime or WinVDIG for Windows machines.

3D

» [OCD](#)

by [Kristian Linn Damkier](#)

The Obsessive Camera Direction (OCD) library allows intuitive control and creation of Processing viewport Cameras.

» [Collada Export](#)

by Marcus Wendt

Exports 3D geometry and material data in the COLLADA format.

» [proSVG](#)

by [Christian Riekoff](#)

Exports vector graphics in the .SVG format.

» [SimplePostScript](#)

by [Marius Watz](#)

Writing vector files in the PostScript format.

» [Compilation](#)

» [unlekker](#)

by [Marius Watz](#)

Contains code export, Merge random numbers, TileSaver class, images from tiling the view, line intersection, complex 2D

» [ImageAdjuster](#)

by [Dave Bellinger](#)

Performs brightness, contrast, gamma and other such adjustments on images. Contains both high-level and low-level routines, and can be extended via user-defined transformations.

» [Data \ Pro](#)

» [Yahoo! Screen](#)

by [Daniel Shiffman](#)

Access the Yahoo! Screen directly in Processing.

» [SFTP](#)

by [Daniel Shiffman](#)

Allows Processing to use SFTP for secure file transfer; uses JSch (Java Secure Channel).

» [Switchboard](#)

by [Jeff Crouse](#)

Web services library. Helps in extracting data from services such as Google, Yahoo, Amazon, Del.icio.us, Flickr, and many more.

» [proMidi](#)

by [Christian Riekoff](#)

Allows Processing to send and receive midi information.

» [Gestalt](#)

by d3 and senior pako

Gestalt is an open structured environment, designed to prototype and develop OpenGL- and Java-based sketches and applications.

» [Fog](#)

by [John G.](#)

Simple fog effect for Processing using P3D and OPENGL renderers.

» [Vector 3D](#)

by [Dan Shiffman](#)

The Vector3D class from The Nature of Code course.

» [Simulation \ Math](#)

» [Physics](#)

by [Jeffrey Traer Bernstein](#)

Simple particle system physics engine. No collisions, just particles, springs, gravity & drag.

» [A* Libraries](#)

by [Aaron Sted](#)

A set of libraries to assist with artificial programming tasks such as genetic algorithms and the AStar algorithm.

» [Cell Noise](#)

by [Carl-Johan Rosén](#)

Explores cell noise (Worley noise), a pattern generation algorithms useful for animation.

» [Oscillator](#)

by [taka](#)

Useful for generating series of waves.

» [Apple SMS](#)

by [Daniel Shiffman](#)

Interface to the Apple Sudden Motion Sensor in PowerBooks (and MacBooks) since 2005.

» [Apple Light Sensor](#)

by [Martin Reffert](#)

Interface to the Light Sensor in MacBook Pro computers.

» [NXTComm](#)

by [Jorge Cardoso](#)

Allows control of the Lego Mindstorms NXT robots.

» [Animation and Typography](#)

» [Sheatween](#)

by [Eli Zanarin](#) and [Golan Levin](#)

An easy way to animate elements in a variety of ways.

» [NextText](#)

by [Eli Zanarin / Obx Labs](#)

Assists in making dynamic and interactive text-based applications.

» [MovingLetters](#)

by [Ilu](#)

Typeface of moving, jittering letters.

» [proXML](#)

by [Christian Riekoff](#)

Allows Processing to read and write XML files.

» [Google API](#)

by [Tatsuya SAITO](#)

Interface to query web pages through Google search.

» [oscP5](#)

by [Andreas Schlegel](#)

An OpenSound Control (OSC) implementation for Processing.

» [MySQL](#)

by [Florian Jenett](#)

Facilitates communication with a MySQL database.

» [Carnivore](#)

by [RSG](#)

A TCP/UDP packet sniffer library for the Processing programming language.

» [MaxLink](#)

by [Jesse Kris](#)

Enables communication between Processing and Max/MSP 4.5

» [Monomic](#)

by [Jesse Kris](#)

Enables communication between Processing and the monome 40h device.

» [UDP](#)

by [Stephane Cousot](#)

Enables simple UDP communication,

» [matrixlib](#)

by [Francis Blitqi](#)

Helpful code for matrix operations.

» [Interface](#)

by [Andreas Schlegel](#)

Custom GUI elements with the ability to show hide and move while the program is running.

» [procCONTROLL](#)

by [Christian Riekoff](#)

Allows Processing to communicate with control devices like joysticks and joypads.

» [MyGUI](#)

by [Markavain](#)

Implements buttons, radio button, checkboxes, sliders, and scroll bars.

» [Interfasia](#)

by [Brendan Berg](#)

Provides a toolkit of standard interface widgets like text fields, buttons, checkboxes, sliders, etc.

» [ID3](#)

by [Jorge Cardoso](#)

Allows Processing to read the ID3v1.1 tags from MP3 files.

» [QRCode](#)

by [Daniel Schiffman](#)

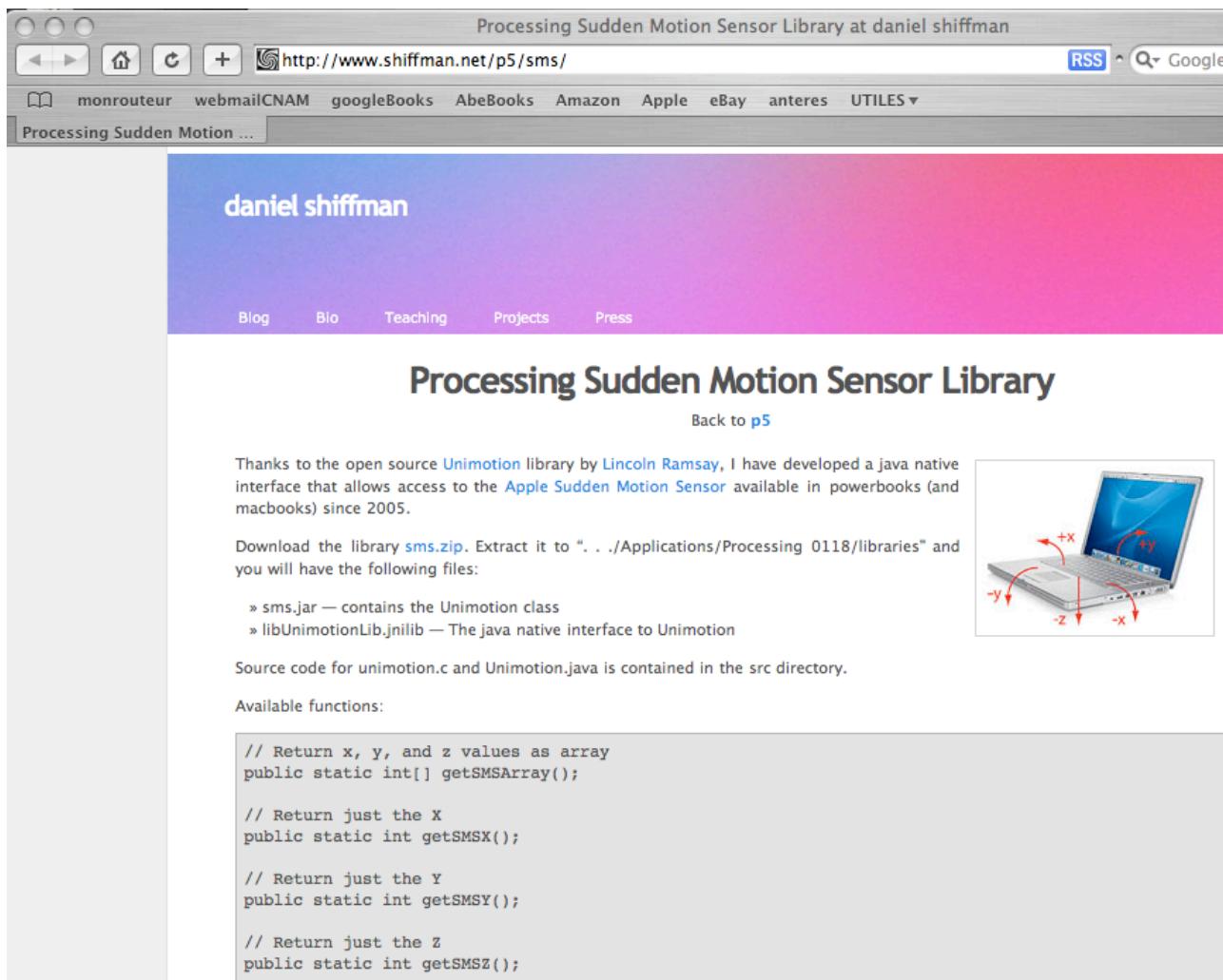
Reads QR Code images, a two-dimensional barcode format.

» [Social Networks Library](#)

by [Tomas Skalicky](#)

Contains network library components, but emphasizes tools for selecting subsets and traversing the network.

Installation typique : exemple de la librairie AppleSMS (SMS = sudden motion sensor)



The screenshot shows a web browser window with the following details:

- Title Bar:** Processing Sudden Motion Sensor Library at daniel shiffman
- Address Bar:** http://www.shiffman.net/p5/sms/
- Toolbar:** Back, Forward, Home, Stop, Refresh, RSS, Google search bar.
- Menu Bar:** monrouteur, webmailCNAM, googleBooks, AbeBooks, Amazon, Apple, eBay, anteres, UTILES ▾
- Page Content:**
 - Header:** daniel shiffman
 - Navigation:** Blog, Blo, Teaching, Projects, Press
 - Title:** Processing Sudden Motion Sensor Library
 - Text:** Thanks to the open source Unimotion library by Lincoln Ramsay, I have developed a java native interface that allows access to the Apple Sudden Motion Sensor available in powerbooks (and macbooks) since 2005.
 - Text:** Download the library [sms.zip](#). Extract it to ".../Applications/Processing 0118/libraries" and you will have the following files:
 - » sms.jar — contains the Unimotion class
 - » libUnimotionLib.jnilib — The java native interface to Unimotion
 - Text:** Source code for unimotion.c and Unimotion.java is contained in the src directory.
 - Text:** Available functions:

```
// Return x, y, and z values as array
public static int[] getSMSArray();

// Return just the X
public static int getSMSX();

// Return just the Y
public static int getSMSY();

// Return just the Z
public static int getSMSZ();
```

Un petit sketch de demo

import

Pb. du
framerate

The screenshot shows the Processing 0.125 Beta interface. The title bar says "Processing - 0125 Beta". The top menu bar includes a play button, file operations, and a "Run" button. The code area contains the following sketch:

```
tiltmacbook2
import sms.*;
int ox, oy, oz;
float ax, ay, az;
int[] vals;
float macc,lacc;
float yaw,pitch,roll;

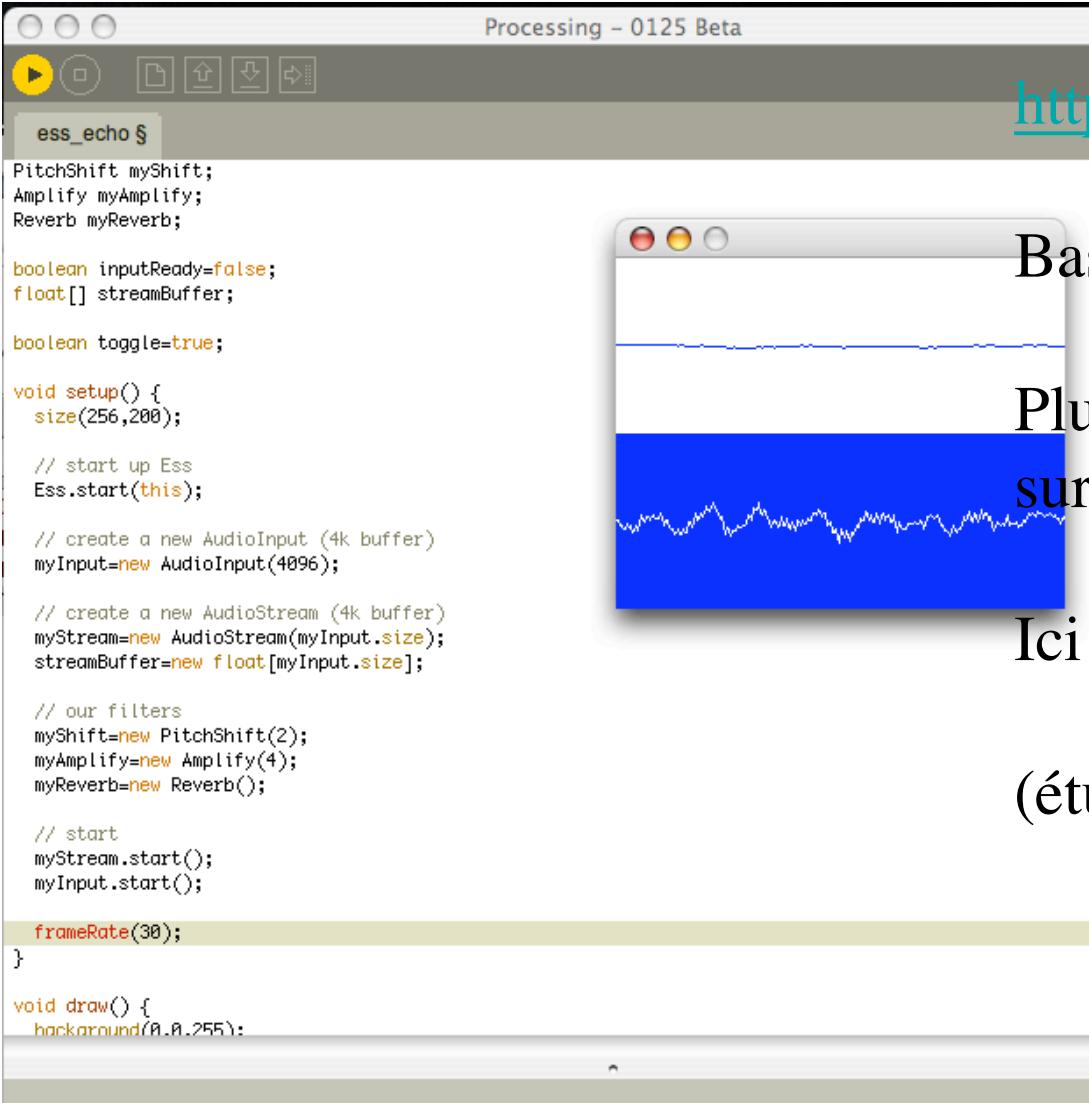
void setup() {
  size(200, 600, P3D);
  ox = oy = oz = 0;
}

void draw() {
  vals = Unimotion.getSMSArray();
  //accelerations en g
  ax = (vals[0]-21)/256.0;
  ay = (vals[1]+17)/256.0;
  az = (vals[2]-11)/256.0;
  //tilt angles
  pitch = atan2(sqrt(ay*ay+az*az),ax)*180/PI;
  roll = atan2(sqrt(ax*ax+az*az),ay)*180/PI;
  //yaw = atan2(sqrt(ax*ax+ay*ay),az)*180/PI;
  yaw = atan2(cos(az),sqrt(1-cos(az)*cos(az)))*180/PI;
  //println(ax+" "+ay+" "+az);
  println("angles : "+pitch+" "+roll+" "+yaw);
  lacc = sqrt(ax*ax+ay*ay+az*az);
  background(255, 204, 0);
  ellipse(100,100,pitch);
  ellipse(100,300,roll,90);
  ellipse(100,500,100*lacc,100*lacc);
  //delay(250);
}
```

The preview window shows a yellow square with three white circles arranged vertically in the center. The bottom-left corner of the preview window has a small black box with the number "1". The bottom-left panel of the interface displays the printed output from the `println` statements in the code:

angles : 88.6831	89.78053	31.585161
angles : 88.46371	89.78055	31.585161
angles : 88.463745	89.56111	31.585161
angles : 88.67293	89.77884	32.032787
angles : 88.68313	89.56107	31.585161
angles : 88.68815	89.562744	31.36135
angles : 88.6831	89.78053	31.585161

Aute ex. : la librairie sonore ESS



```
Processing - 0125 Beta
ess_echo $
```

```
PitchShift myShift;
Amplify myAmplify;
Reverb myReverb;

boolean inputReady=false;
float[] streamBuffer;

boolean toggle=true;

void setup() {
    size(256,200);

    // start up Ess
    Ess.start(this);

    // create a new AudioInput (4k buffer)
    myInput=new AudioInput(4096);

    // create a new AudioStream (4k buffer)
    myStream=new AudioStream(myInput.size());
    streamBuffer=new float[myInput.size()];

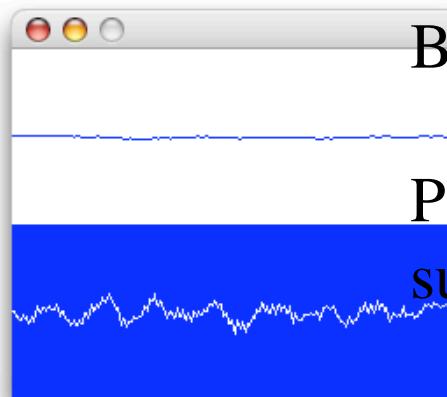
    // our filters
    myShift=new PitchShift(2);
    myAmplify=new Amplify(4);
    myReverb=new Reverb();

    // start
    myStream.start();
    myInput.start();

    frameRate(30);
}

void draw() {
    background(0,0,255);
```

<http://www.tree-axis.com/Ess/>



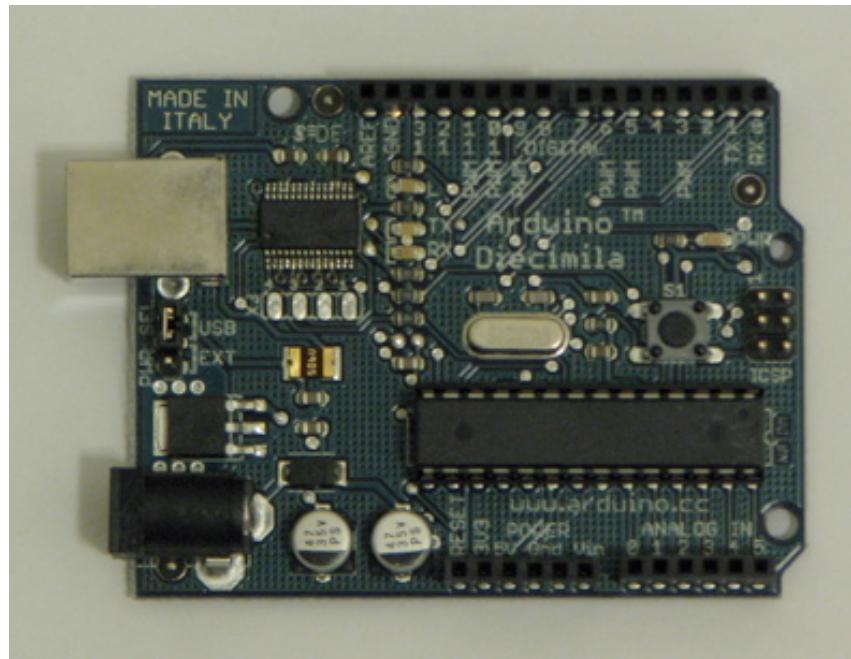
Basée sur JavaSound

Plusieurs très bons ex.
sur le site

Ici : squirrel-echo

(étudier ce code !)

5. Les projets « fils » (sisters !)



<http://www.arduino.cc>