

NSY116 - 2008

Introduction à Processing

P. Cubaud <cubaud@cnam.fr>

1. L'environnement
2. Le langage
3. Les trois styles de programmation
4. Les libraries
5. Des exemples



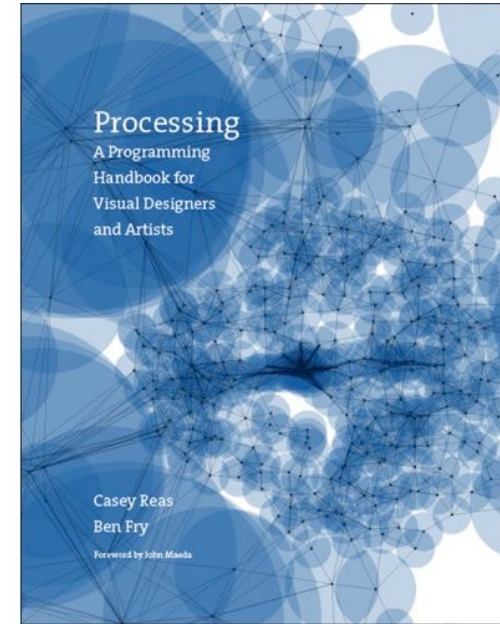
<http://processing.org>

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



<http://reas.com/>
(ucla)

<http://benfry.com/>
(mit medialab)

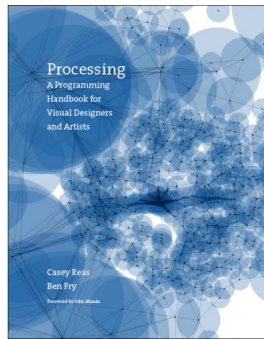


9/2007, 736 p.
MIT Press (35 euros)

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>

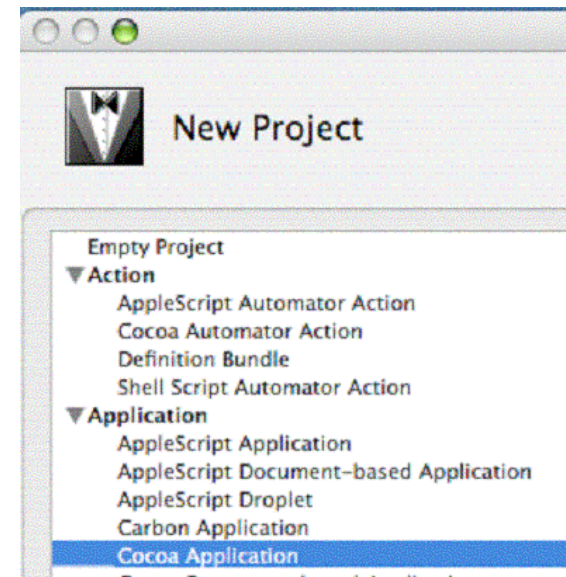
Sketching is necessary for the development of ideas It is necessary to sketch in a medium related to the final medium so the sketch can approximate the outcome. Painters may construct elaborate drawings and sketches before executing the final work. Architects traditionally work first in cardboard and 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)

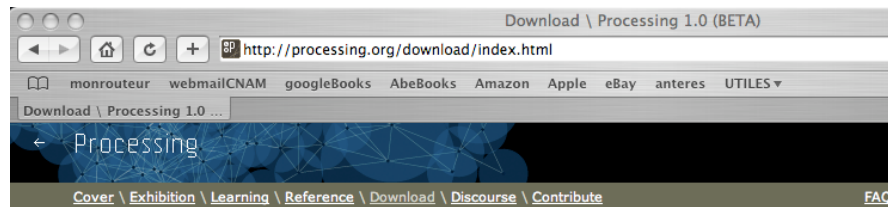


Digression sur Flash



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.



Download Processing (BETA). Processing is available for Linux, Mac OS X, and Windows. Select your choice below to download the software.

THE PROCESSING SOFTWARE IS PROVIDED TO YOU "AS IS," AND WE MAKE NO EXPRESS OR IMPLIED WARRANTIES WHATSOEVER WITH RESPECT TO ITS FUNCTIONALITY, OPERABILITY, OR USE, INCLUDING, WITHOUT LIMITATION, ANY IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, OR INFRINGEMENT. WE EXPRESSLY DISCLAIM ANY LIABILITY WHATSOEVER FOR ANY DIRECT, INDIRECT, CONSEQUENTIAL, INCIDENTAL OR SPECIAL DAMAGES, INCLUDING, WITHOUT LIMITATION, LOST REVENUES, LOST PROFITS, LOSSES RESULTING FROM BUSINESS INTERRUPTION OR LOSS OF DATA, REGARDLESS OF THE FORM OF ACTION OR LEGAL THEORY UNDER WHICH THE LIABILITY MAY BE ASSERTED, EVEN IF ADVISED OF THE POSSIBILITY OR LIKELIHOOD OF SUCH DAMAGES.

By downloading the software from this page, you agree to the specified terms.

Announcements

Email address

Submit

If you are interested in receiving updates about Processing, submit your email through this form. Your email will only be used to send infrequent updates about Processing. It will not be sold or shared.

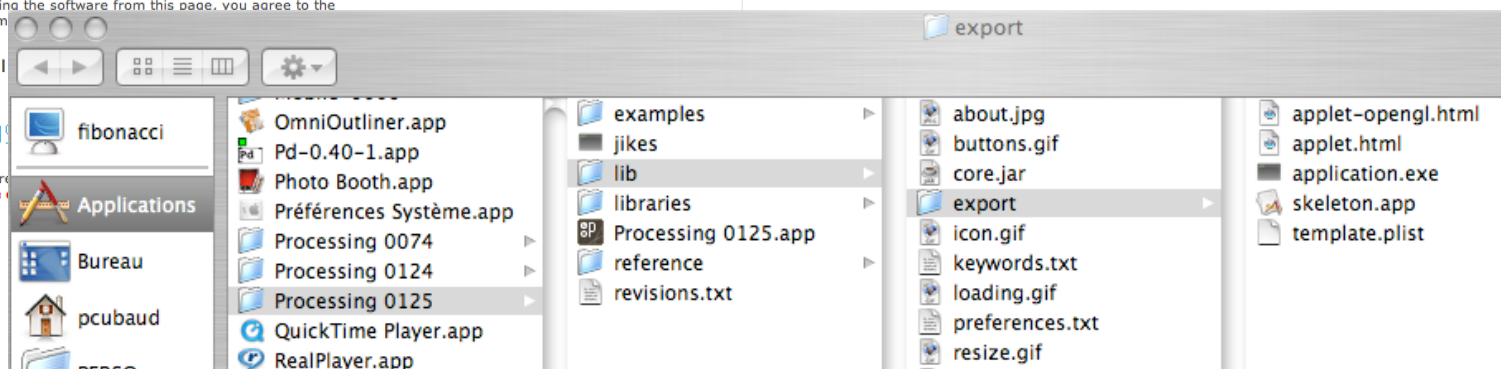
0133 BETA |

↓ Linux

↓ Mac OS

Changes for re

Read before



REV 0068 - 2 février 2004

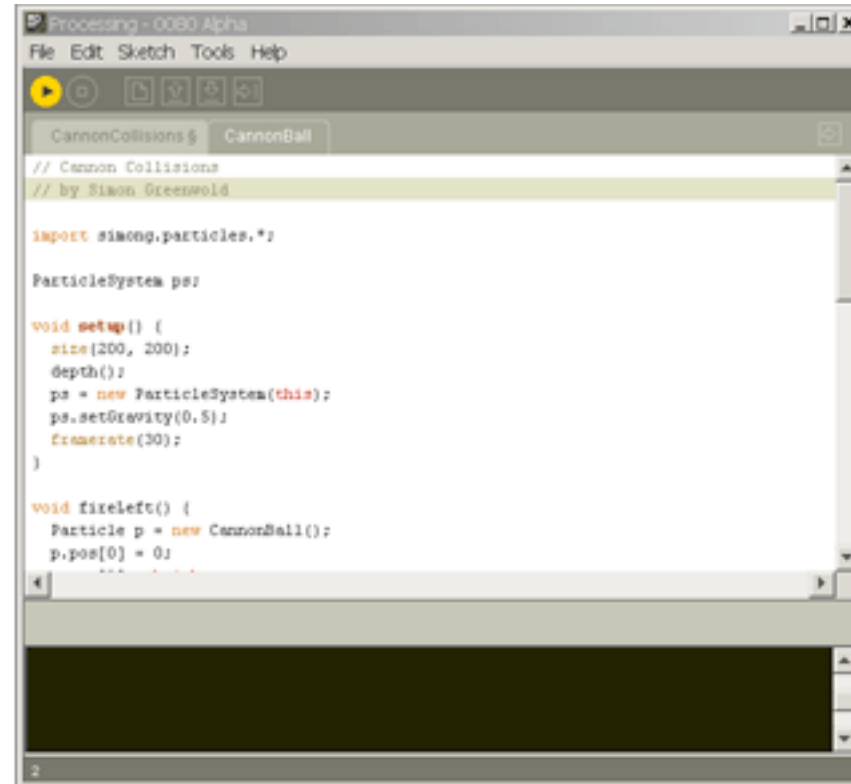
REV 0133 - 26 octobre 2007

Et toujours en version Beta pour la 1.0 !!

Processing Development Environment (PDE)



Display Window



Menu

Toolbar

Tabs

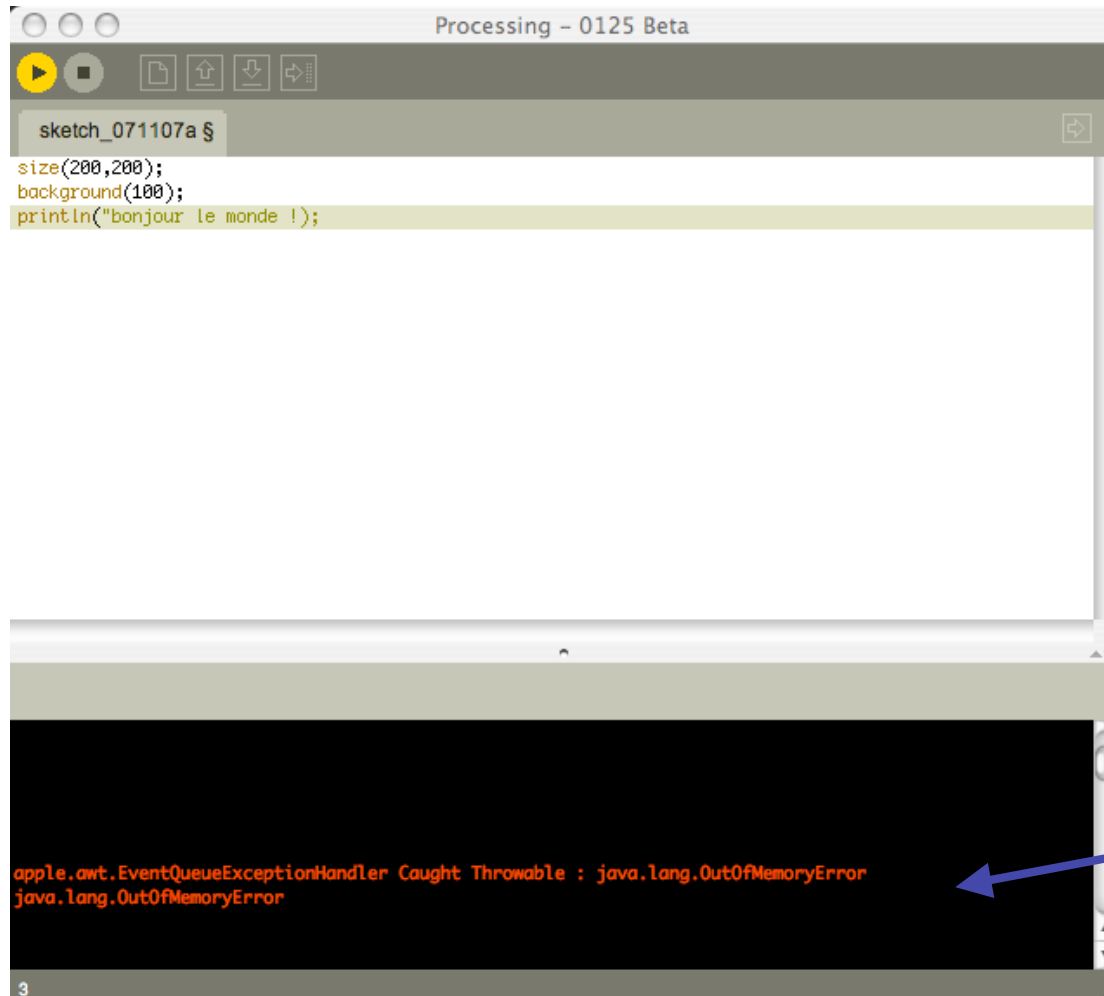
Text Editor

Message Area

Text Area

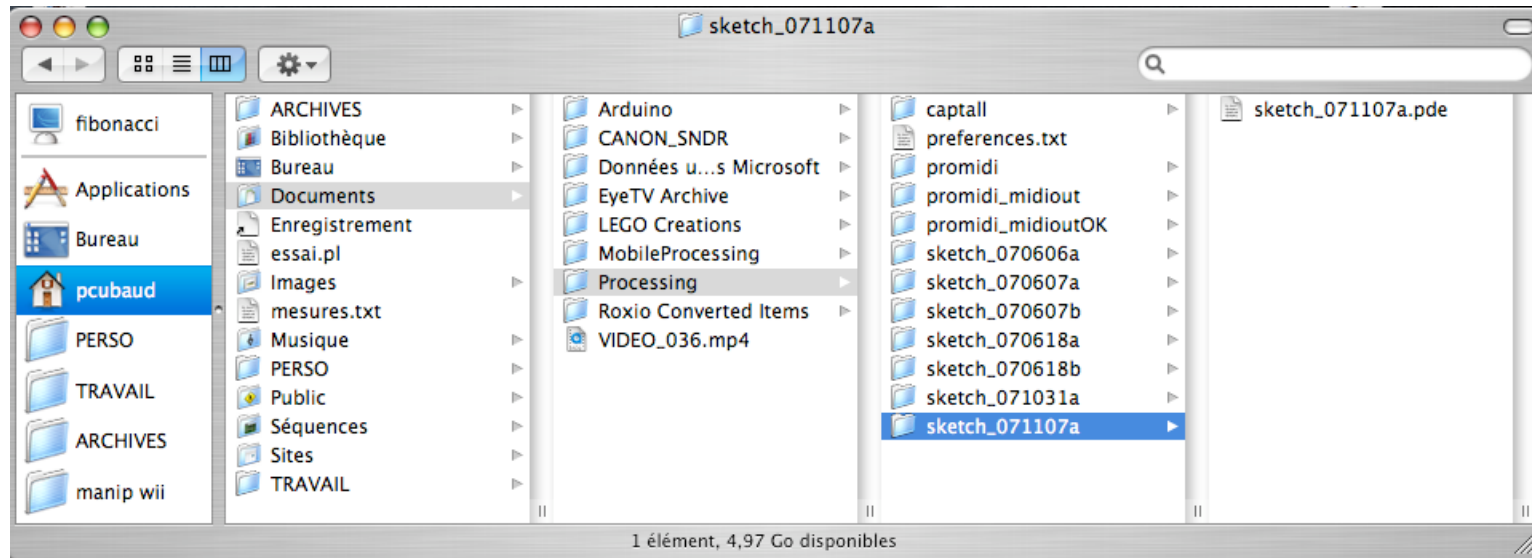
<http://processing.org/reference/environment/index.html>

Bonjour Monde

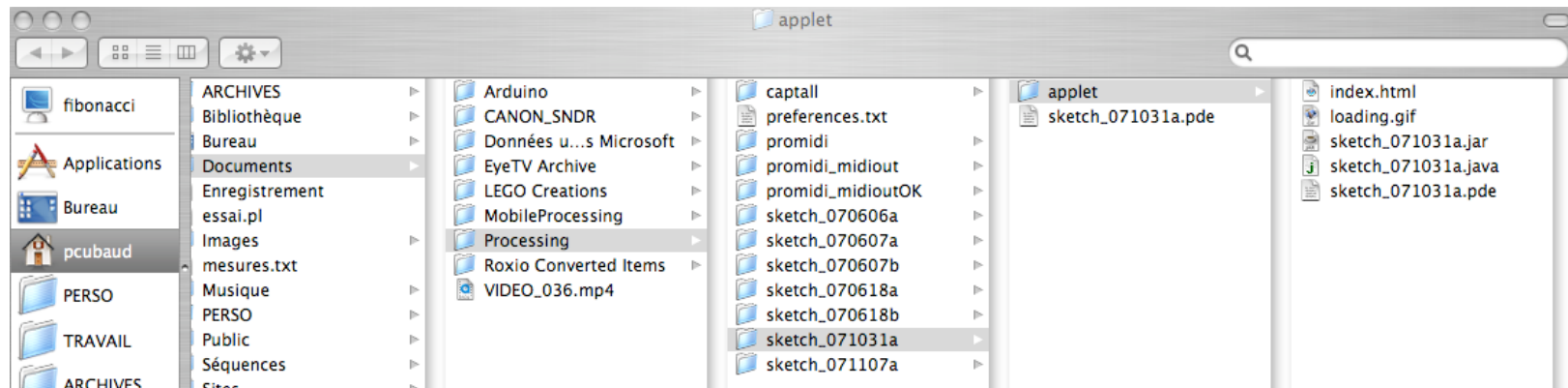


hum...

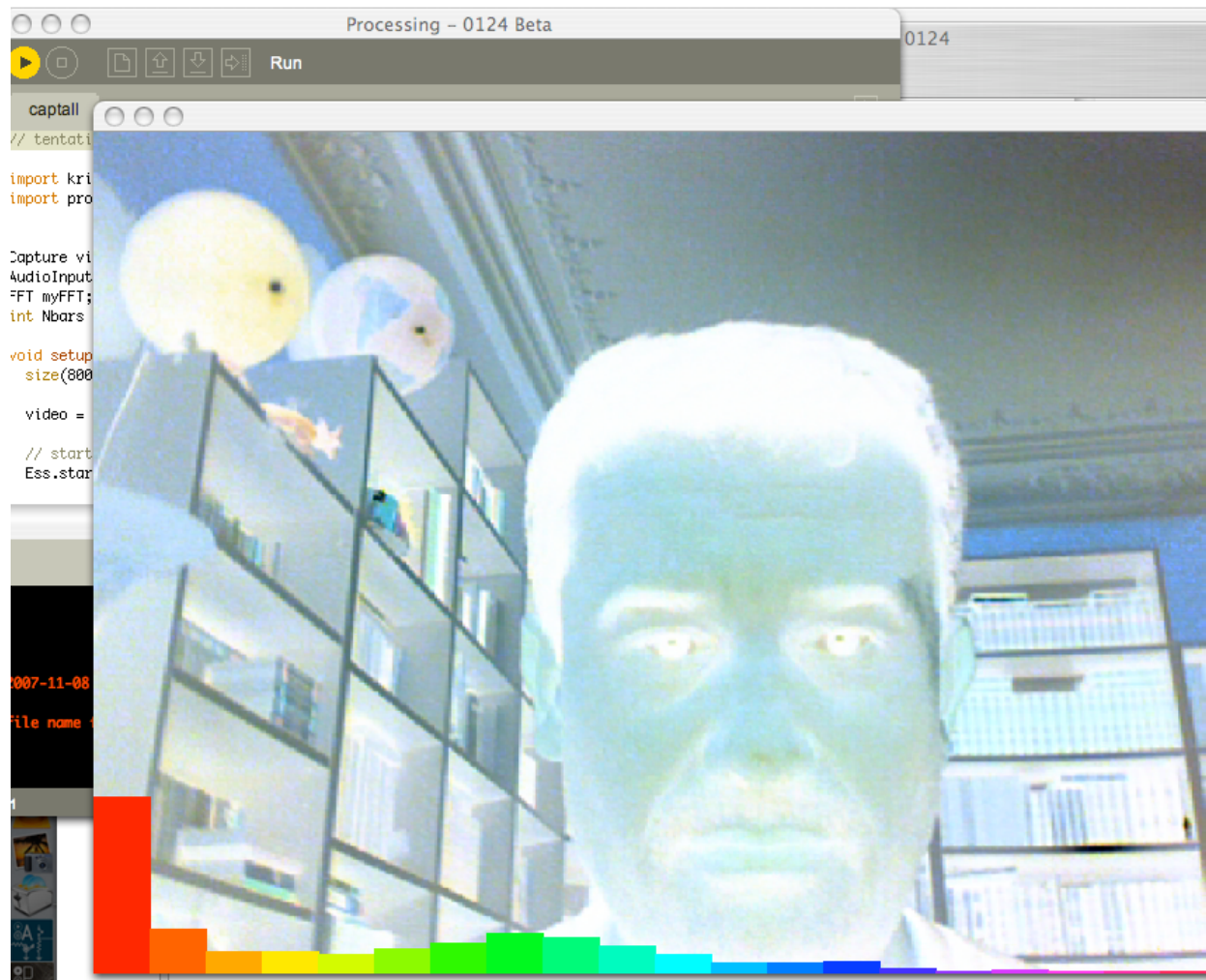
Après la sauvegarde : on crée un fichier .pde rangé par défaut dans le dossier « Documents »



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);
}
```

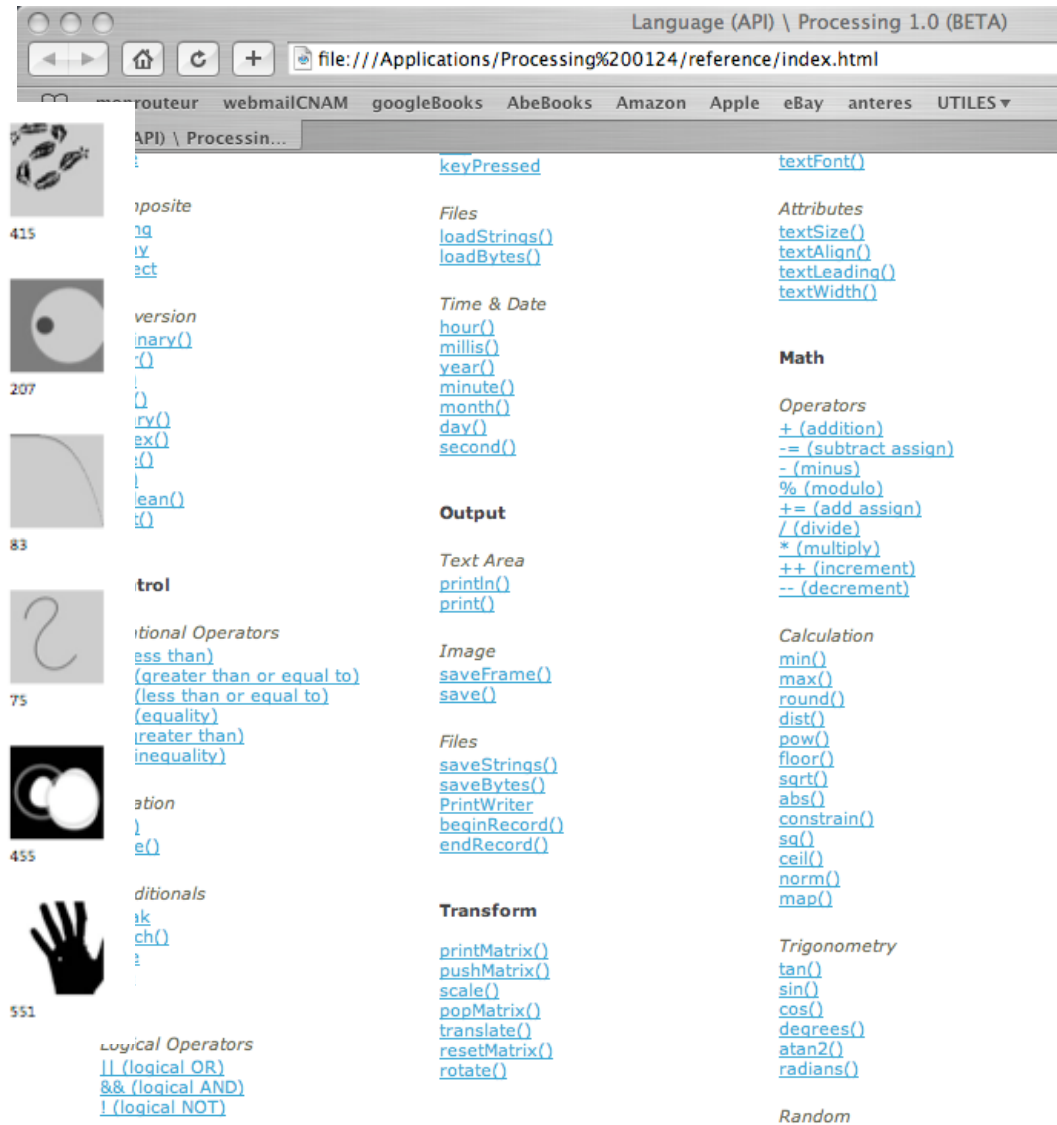
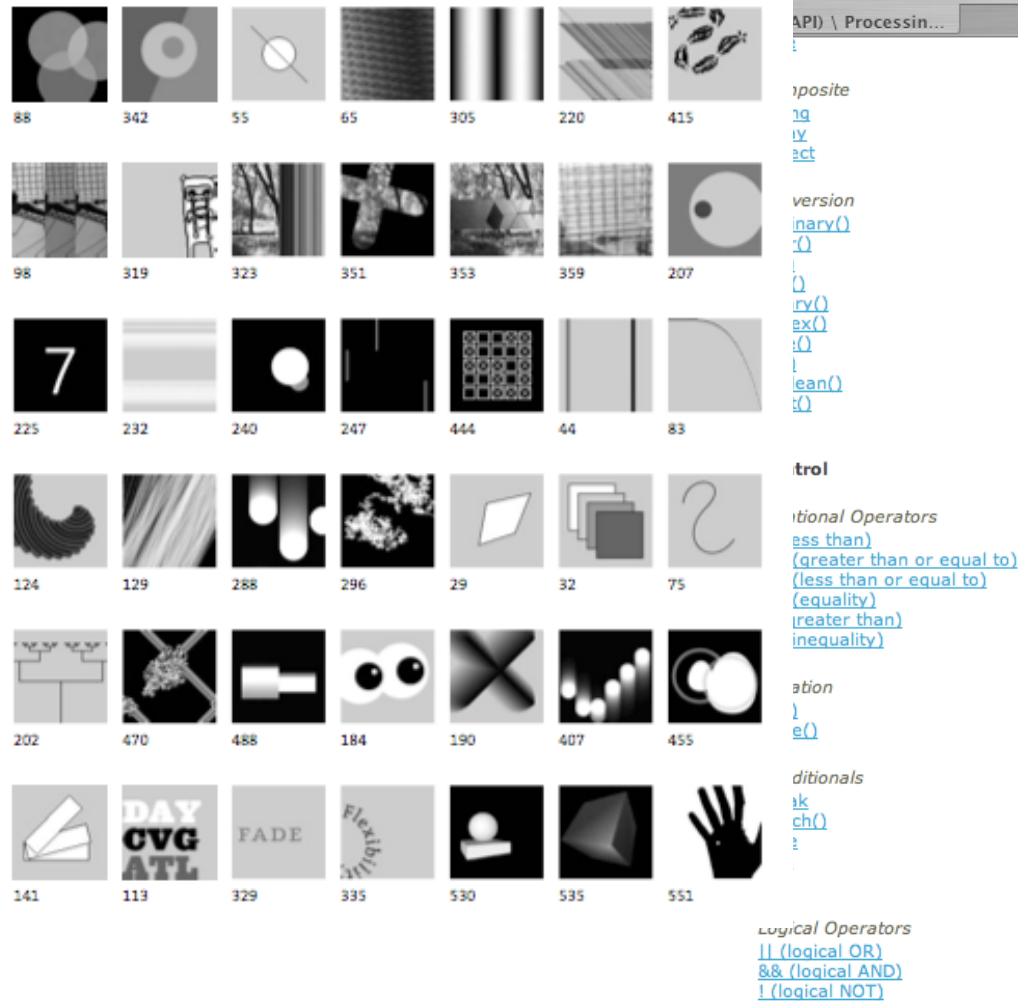
```
public void stop() {
  Ess.stop();
  super.stop();
}
```

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

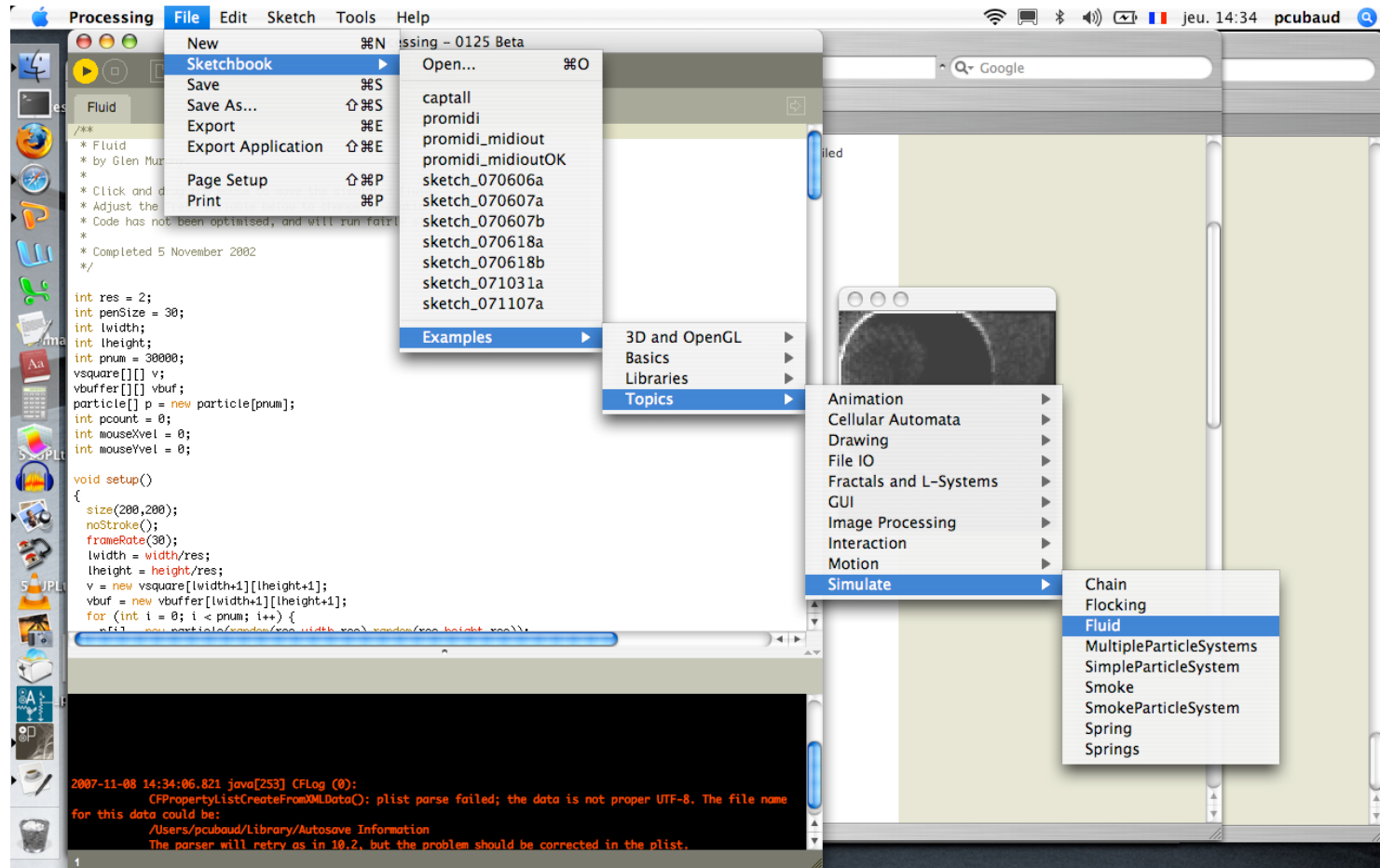
2. Le langage (par l'exemple...)

Manuel de ref.

dans le livre



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



(*) `cd /Applications/Processing\ 0125/examples/ ;
ls -lR | grep pde | wc -l`

Processing par rapport à Java

Input

```
mouseX  
mouseY
```

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

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

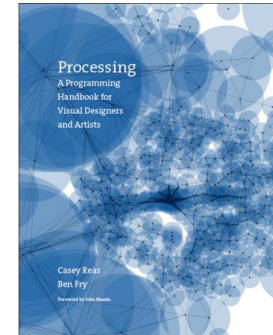
```
/* Assuming there are two variables in  
the program named mouseX and mouseY,  
these values must be changed by the  
programmer in the mouseMoved() and  
mouseDragged methods. */
```

```
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.

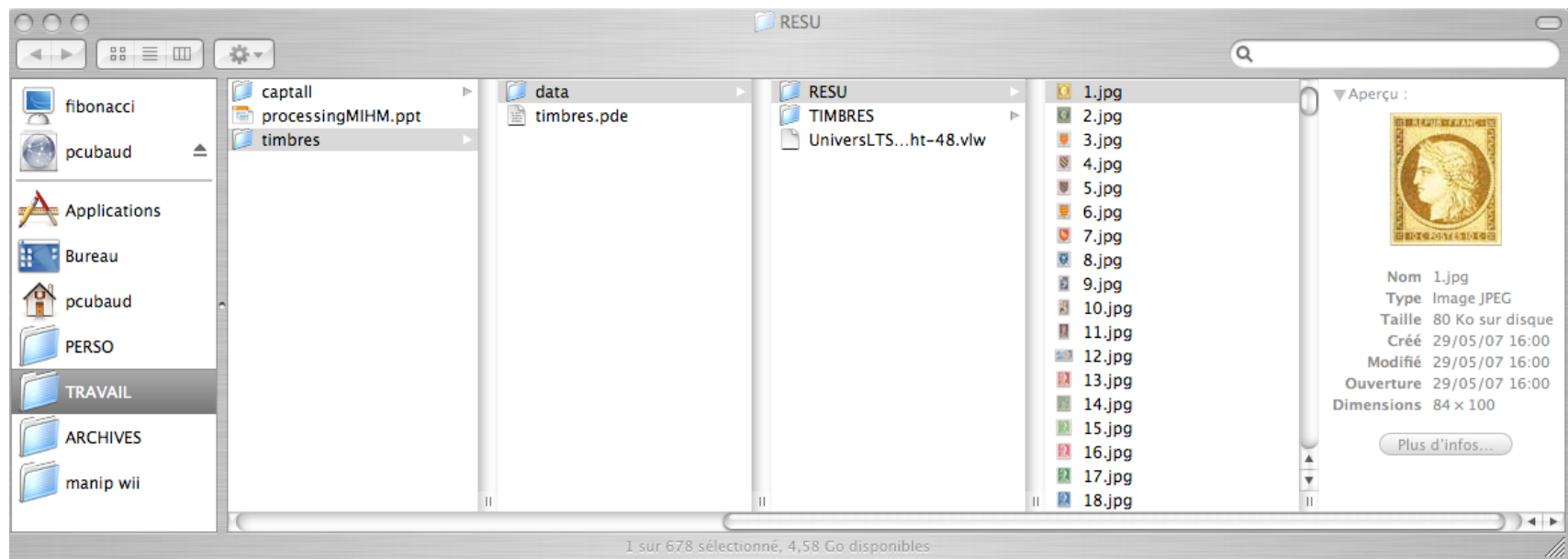
This additional structure allows writing custom functions and classes and using keyboard and mouse events.

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 imagerie de timbres

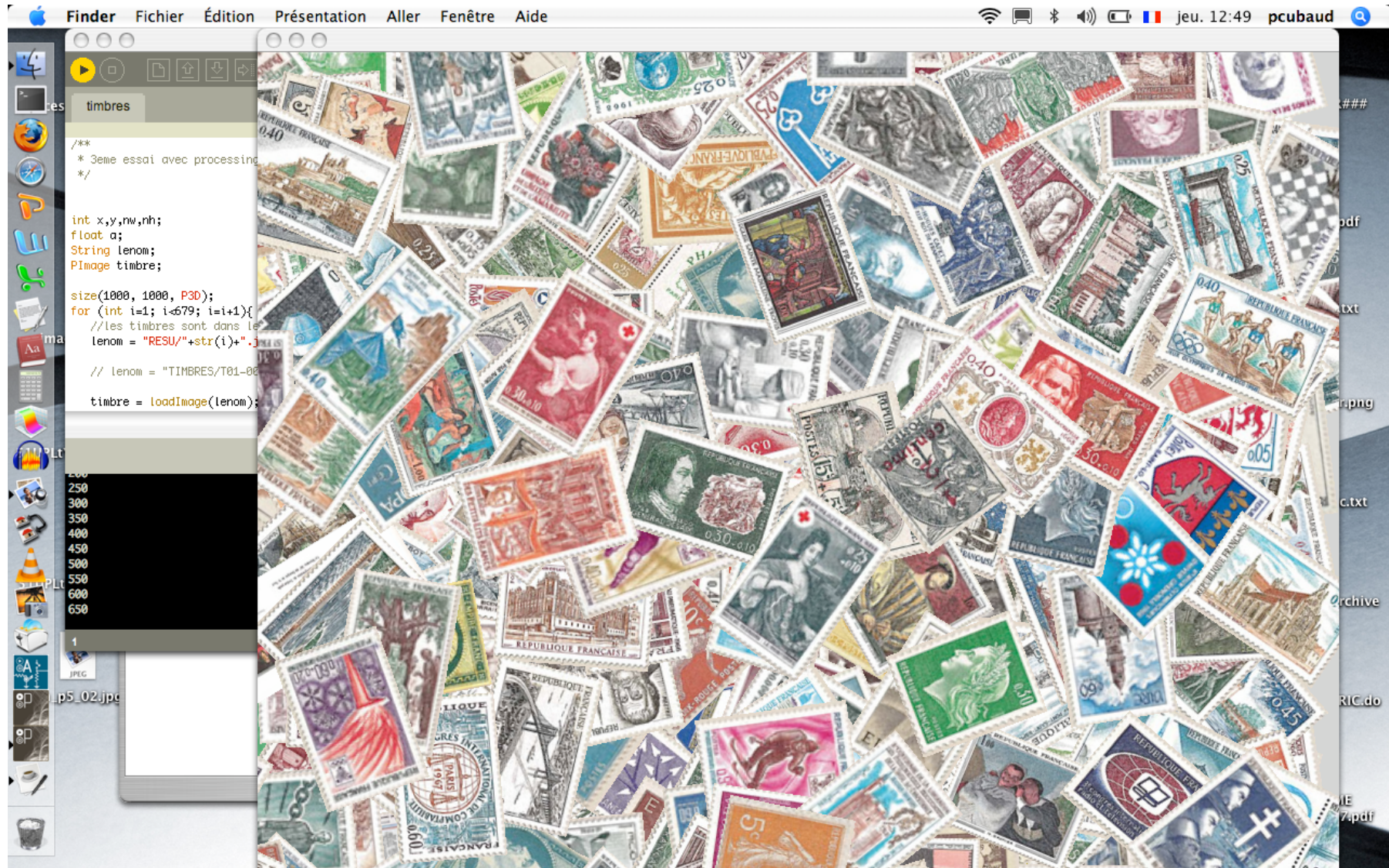
Le code

(commenter)

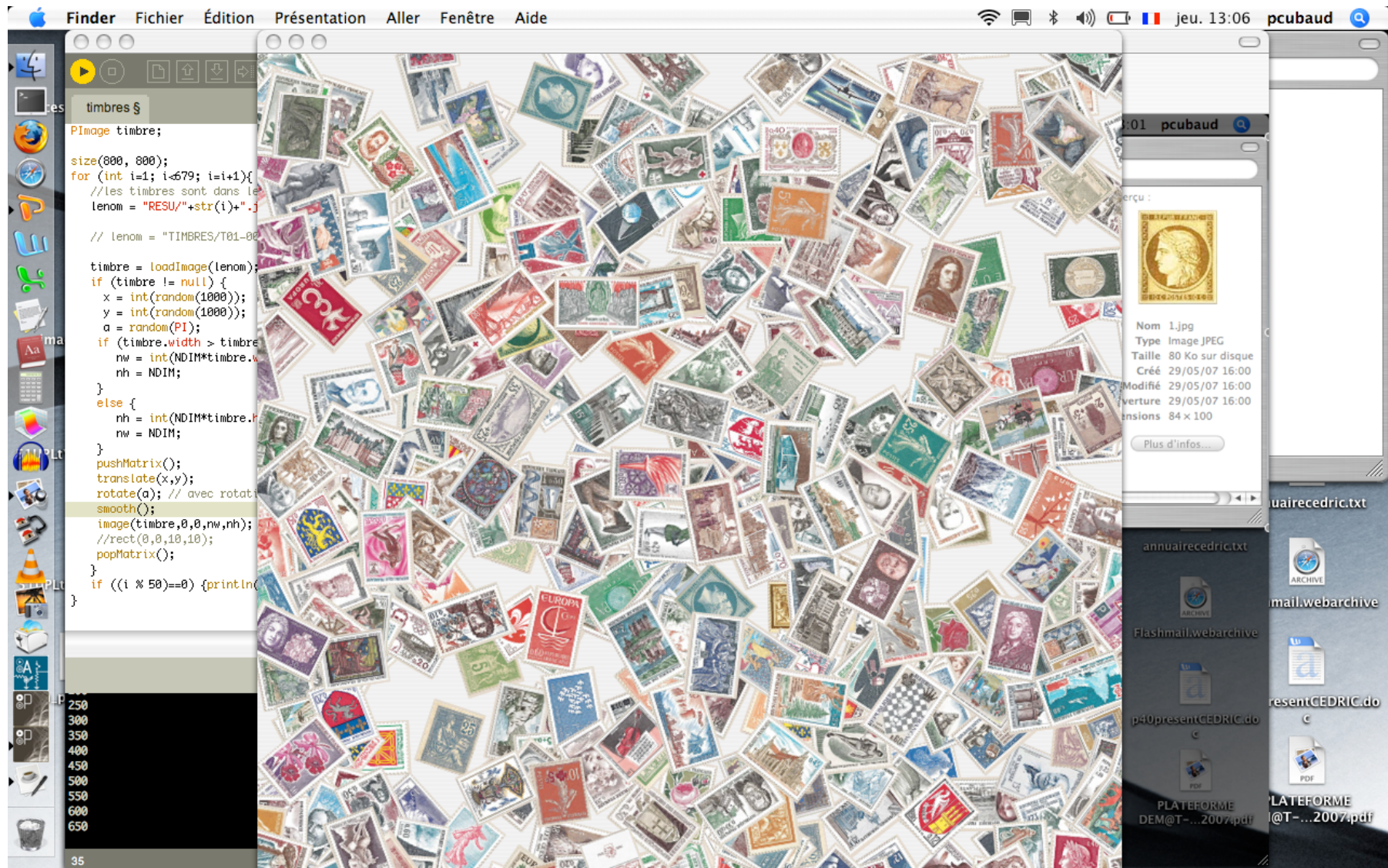
```
Int NDIM = 100;
int x,y,nw,nh;
float a;
String lenom;
PImage timbre;

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);}
}
```

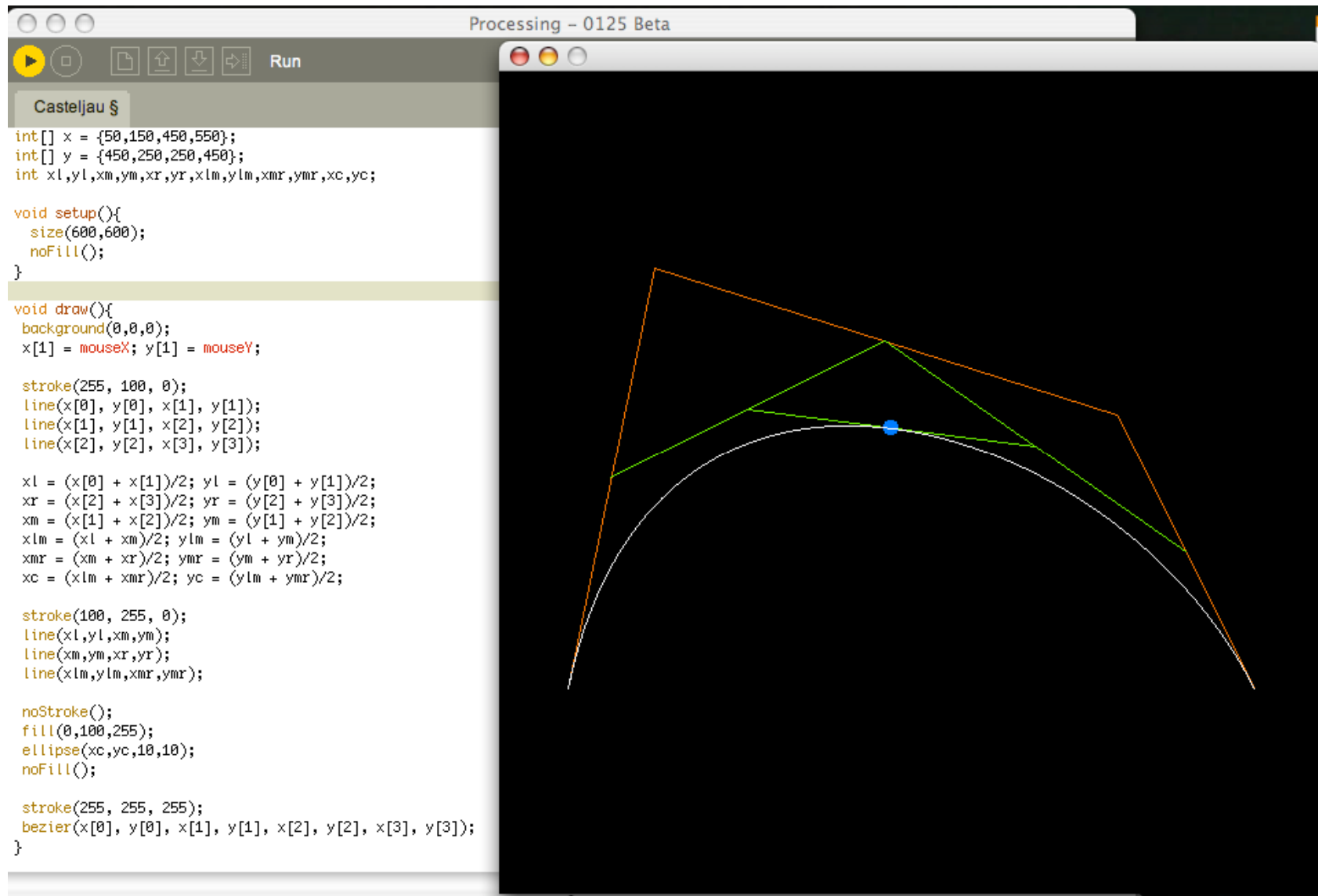
Résultat (NDIM=100)



Résultat (NDIM=50)



3.2 Exemple en mode continuous (algo. de De Casteljau)

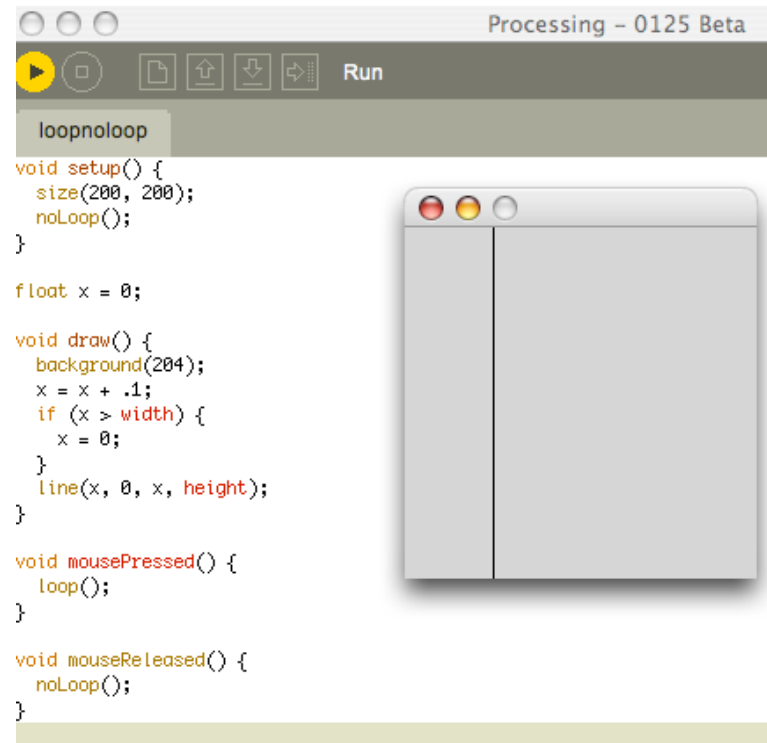


Un autre exemple en mode continuous (extrait du manuel de ref.)

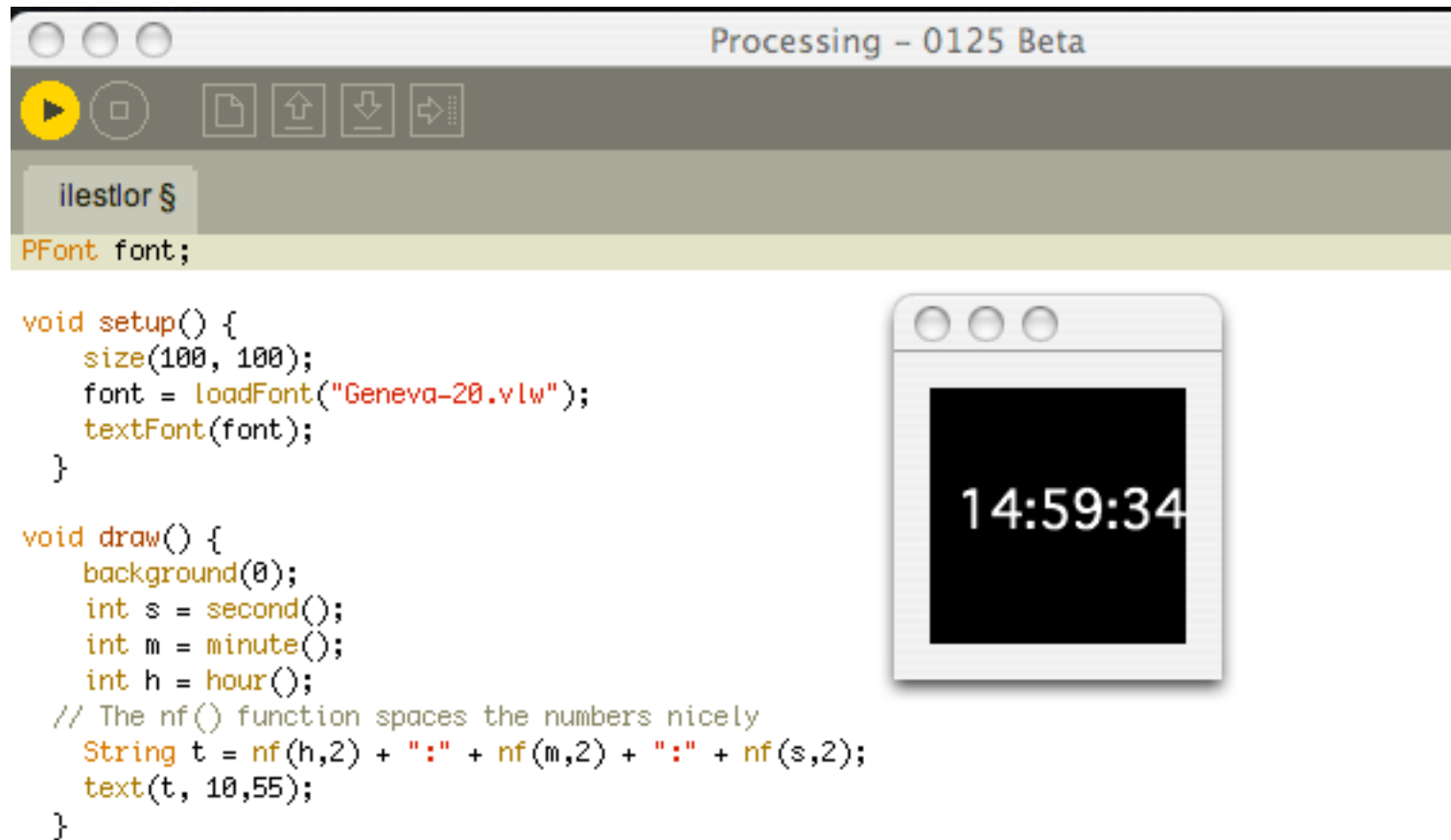
```
void setup() {  
  size(200, 200);  
  noLoop();  
}  
  
float x = 0;  
  
void draw() {  
  background(204);  
  x = x + .1;  
  if (x > width) {  
    x = 0;  
  }  
  line(x, 0, x, height);  
}
```

```
void mousePressed() {  
  loop();  
}
```

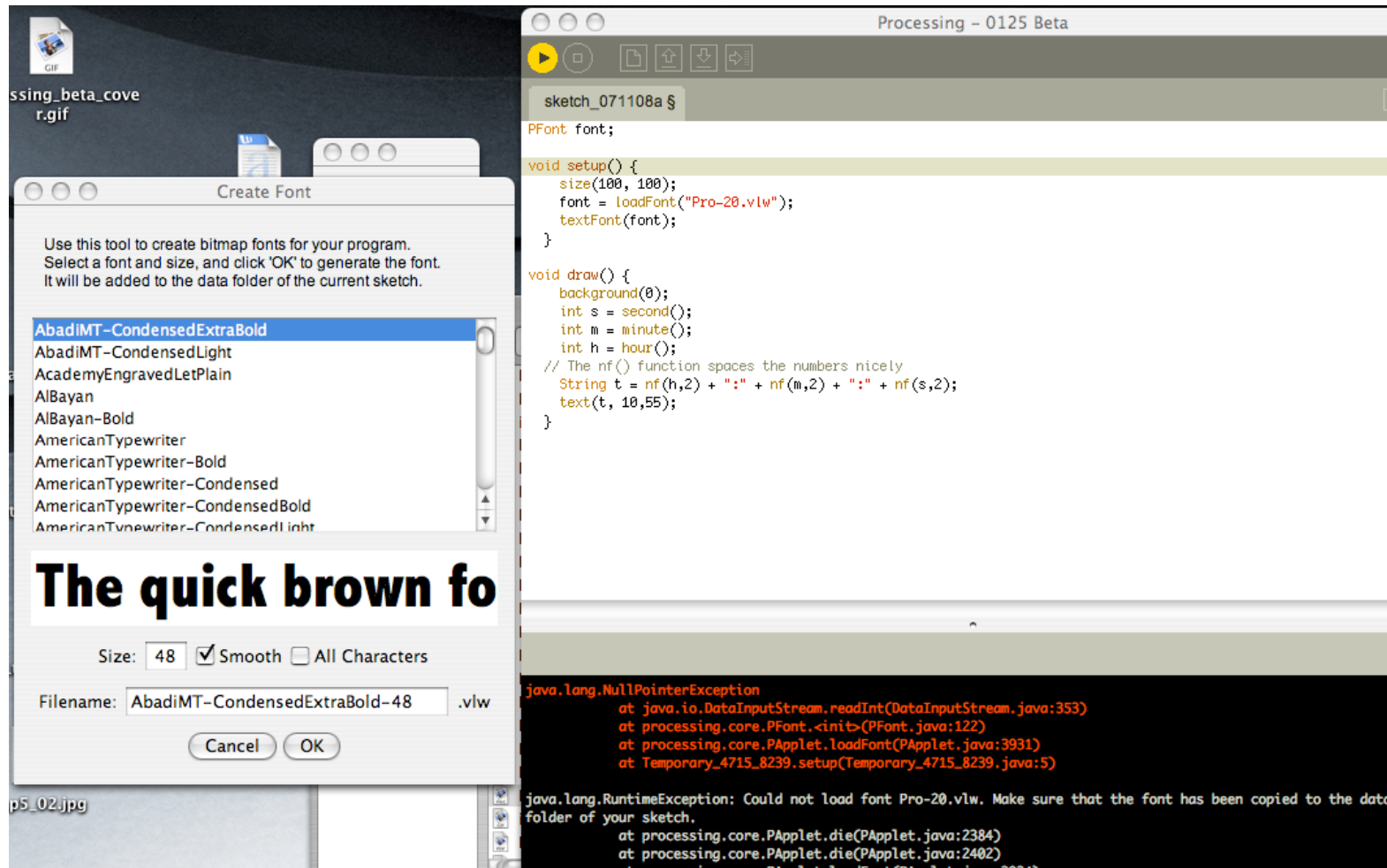
```
void mouseReleased() {  
  noLoop();  
}
```



Un autre exemple avec gestion du temps



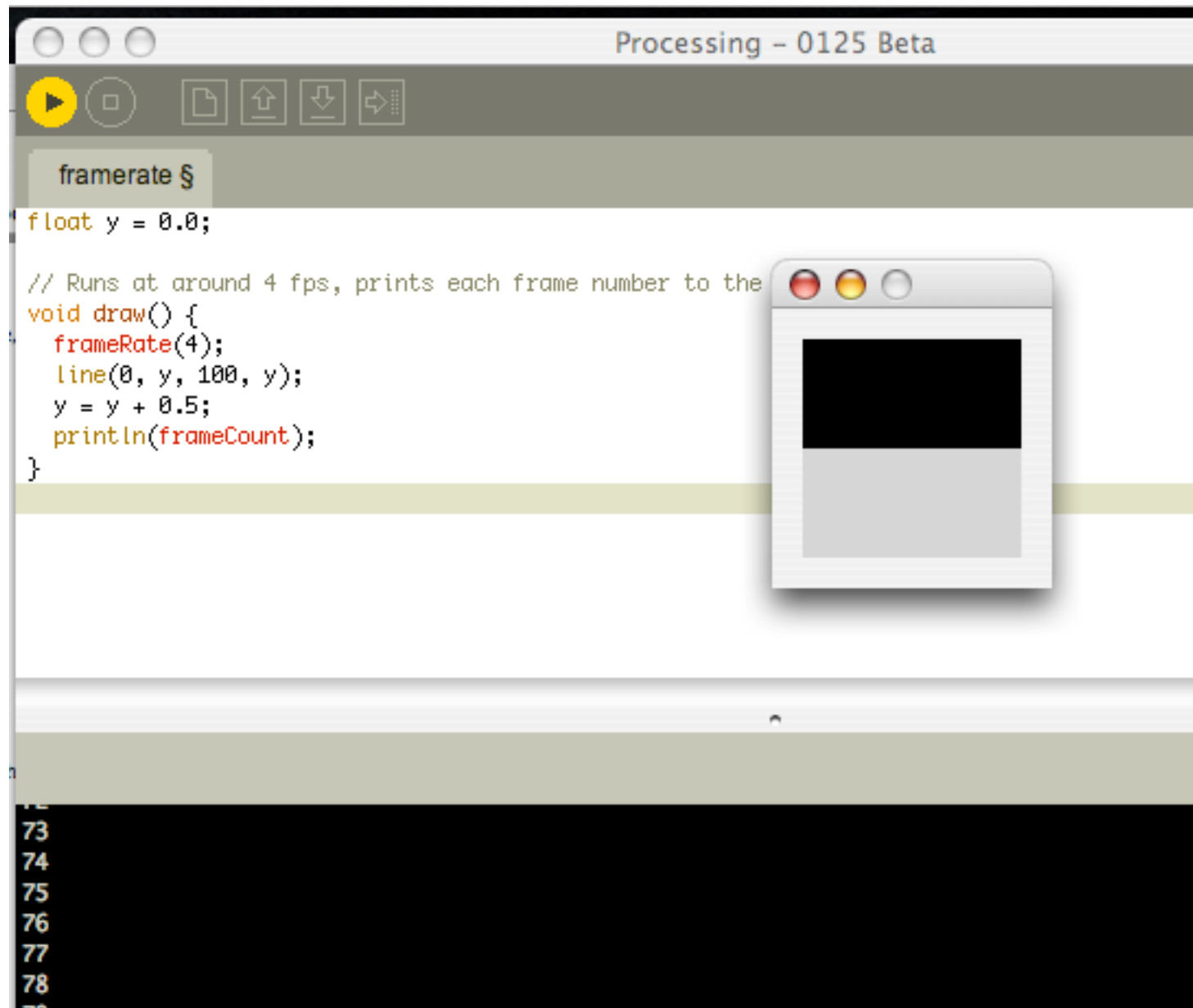
Digression sur les fontes



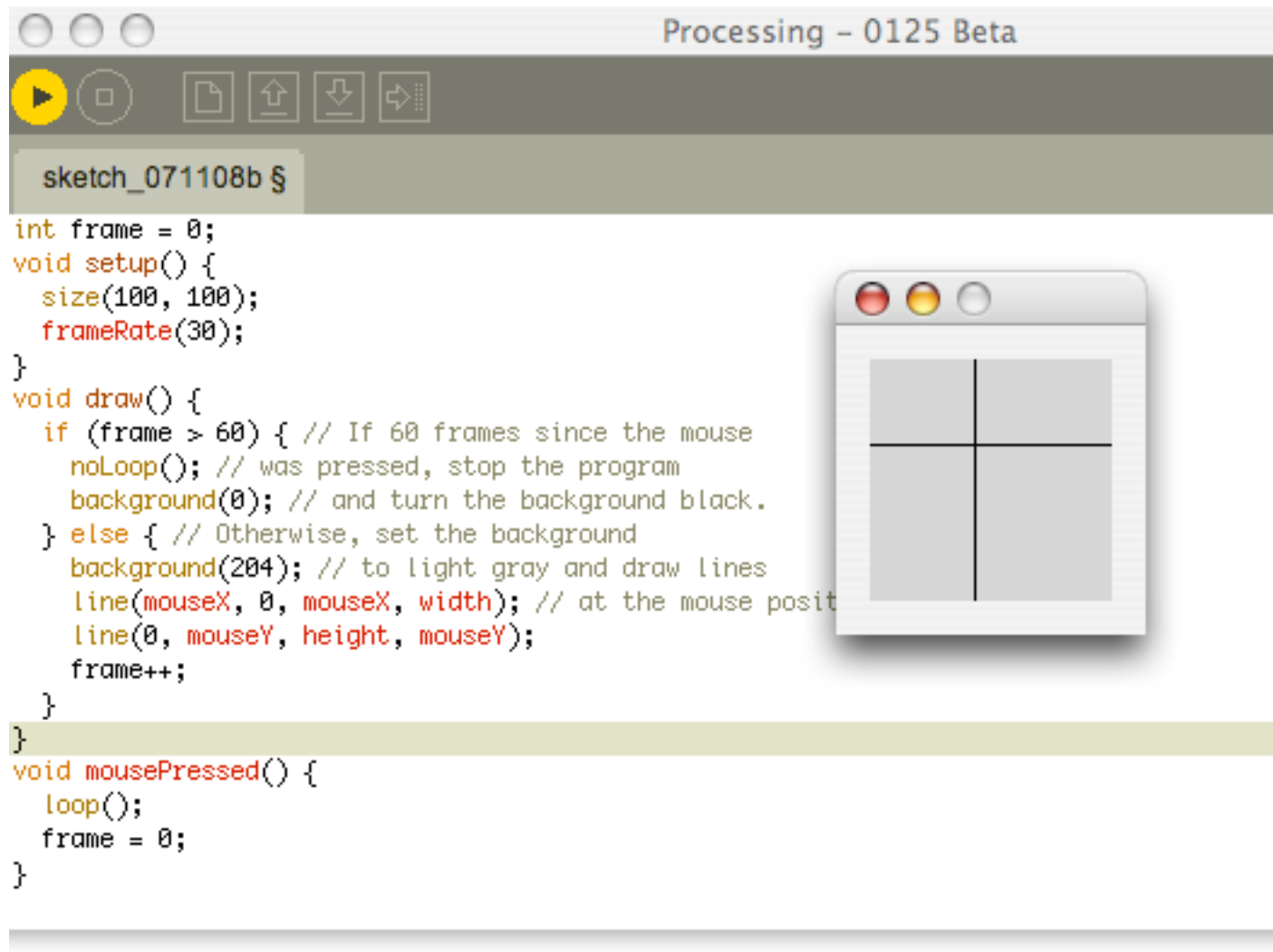
Générées par un outil de l'env.

Ajoutées dans un dossier *data* du sketch

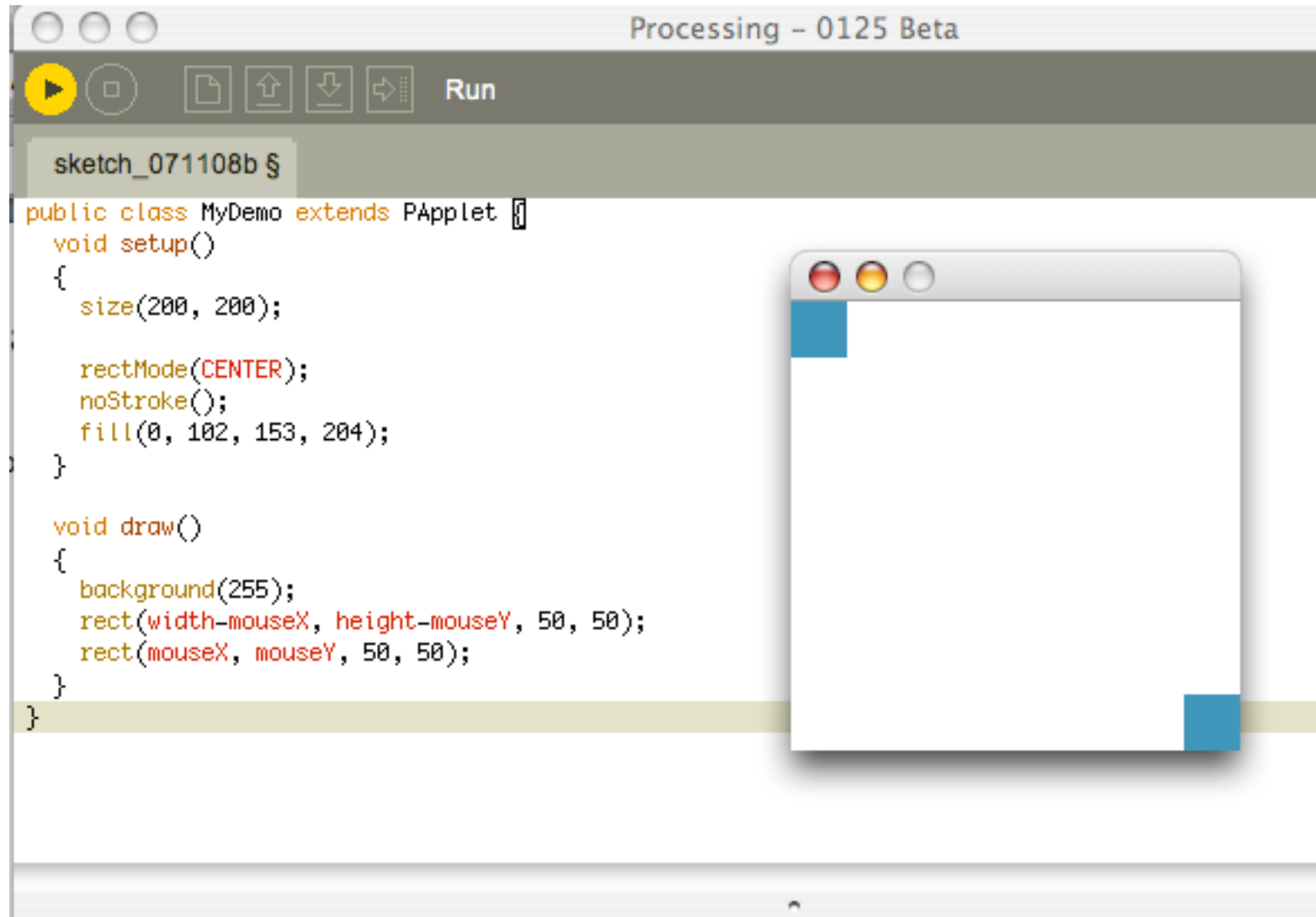
Digression sur le frame-rate



un autre exemple

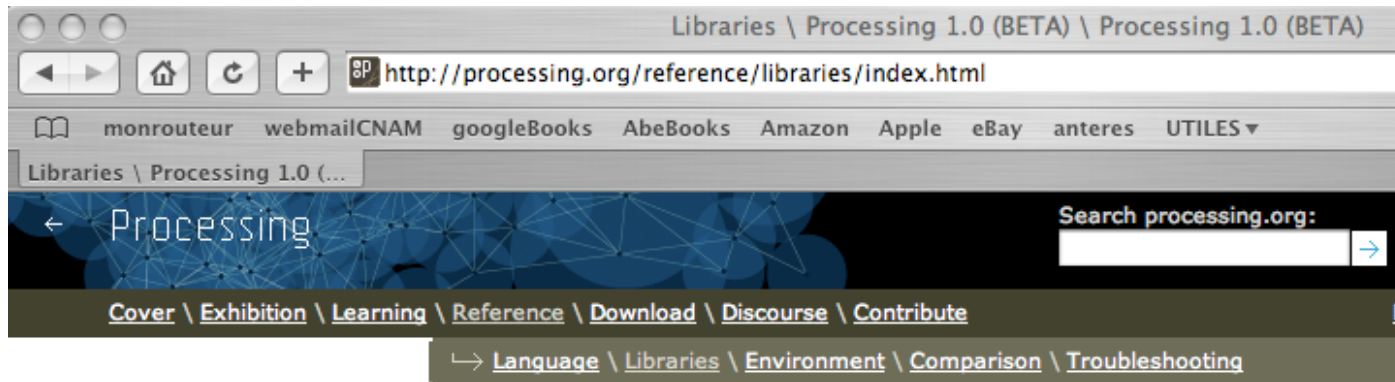


3.3 Exemple en mode java (manuel de ref.)



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.

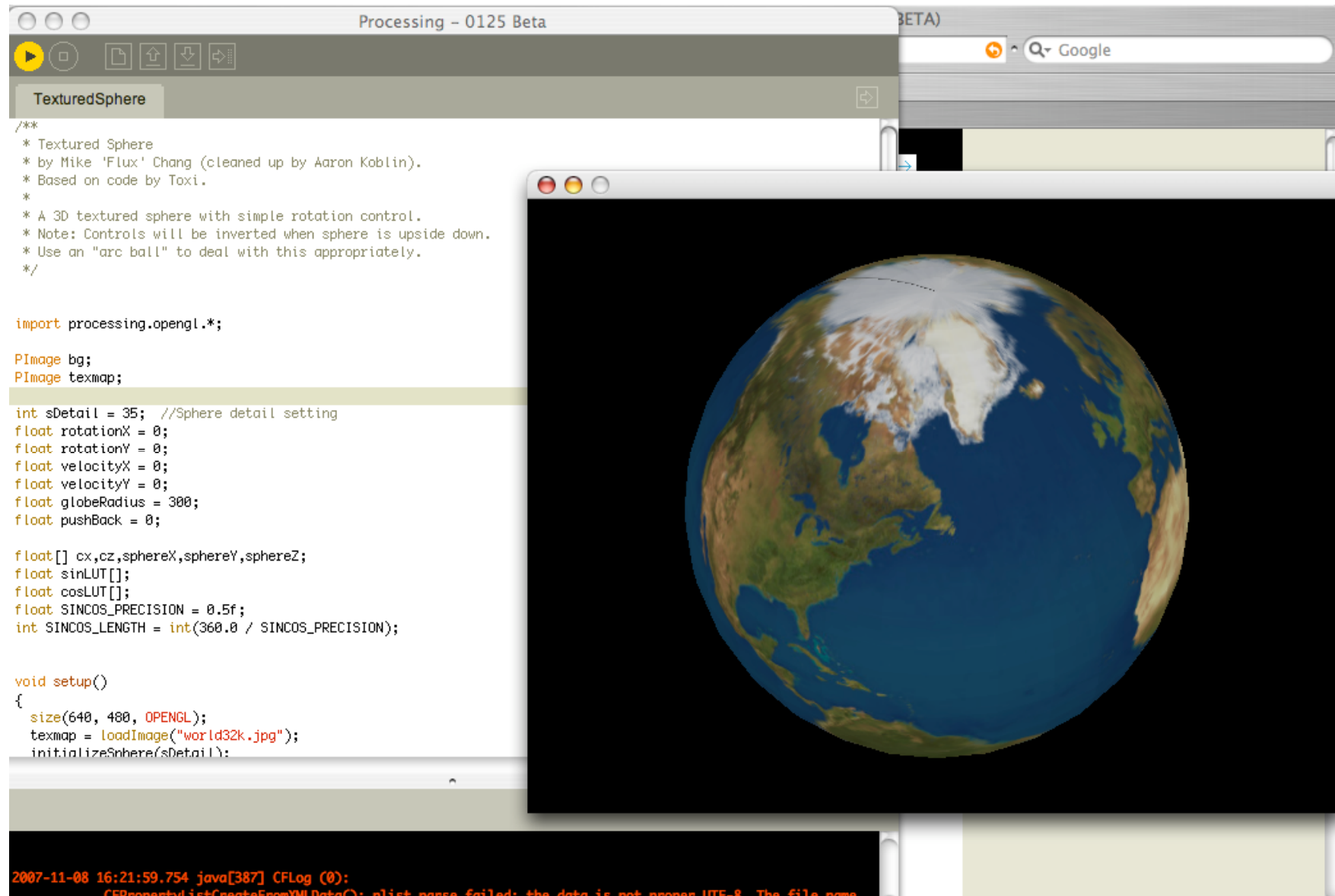
[DXF Export](#)

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



La librairie vidéo : ex. du « tracking »



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, glob distinction, and pixel addressing. Does not require QuickTime or WinVDIG for Windows machines.

3D

» [OCD](#)

by [Kristian Linn Damkjer](#)
The Obsessive Camera Direction (OCD) library allows intuitive control and creation of Processing viewport Cameras.

Compilation

» [unlekker!](#)

by [Marius W](#)
Contains code export, Mers random number generator, TileSaver class, images from tiling the view, line intersection, complex 2D

Data \ Pro

» [Yahoo! S](#)

by [Daniel Shiffman](#)
Access the Yahoo! API 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.

» [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.

Tools

» [proDOC](#)

by [Christian Riekoff](#)
Generates documentation for Processing libraries from the source code. Requires inserting tags in the comments of the classes, fields, and methods.

» [ImageAdjuster](#)

by [Dave Bollinger](#)
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.

Hardware

» [Most Pixels Ever](#)

by [Daniel Shiffman](#) and Chris Kairalla
Framework for spanning Processing sketches across multiple screens.

» [Gestalt](#)

by d3 and senor pako
Gestalt is an open structured environment, designed to prototype and develop OpenGL- and Java-based sketches and applications.

» [Fog](#)

by [JohnG](#)
Simple fog effect for Processing using P3D and OPENGGL 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.

» [AI Libraries](#)

by [Aaron Steed](#)
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 Rädinger](#)
Interface to the Light Sensor in MacBook Pro computers.

» [NXTComm](#)

by [Jorge Cardoso](#)
Allows control of the Lego Mindstorms NXT robots.

Animation and Typography

» [Shapetween](#)

by [Elio Zanarini](#) and [Golan Levin](#)
An easy way to animate elements in a variety of ways.

» [NextText](#)

by [Elio Zanarini](#) / [Obx Labs](#)
Assists in making dynamic and interactive text-based applications.

» [MovingLetters](#)

by [llu](#)
Typeface of moving, jittering letters.

» [proXML](#)

by [Christian Riekoff](#)
Allows Processing to read and write XML files.

» [Google API](#)

by [Tatsuva SAITO](#)
Interface to query web pages through Google search.

» [oscPS](#)

by [Andreas Schlegel](#)
An [OpenSound Control](#) (OSC) implementation for Processing.

» [MySQL](#)

by [Florian Jenett](#)
Facilitates communication with a MySQL database.

» [Carnivore](#)

by [RSB](#)
A TCP/UDP packet sniffer library for the Processing programming language.

» [MaxLink](#)

by [Jesse Kriss](#)
Enables communication between Processing and Max/MSP 4.5

» [Monomic](#)

by [Jesse Kriss](#)
Enables communication between Processing and the monome 40h device.

» [UDP](#)

by [Stephane Cousot](#)
Enables simple UDP communication.

waves.

» [MatrixMath](#)

by [Francis Blount](#)
Helpful code for matrix operations.

Interface

» [controlIPS](#)

by [Andreas Schlegel](#)
Custom GUI elements with the ability to show/hide and move while the program is running.

» [proCONTROL](#)

by [Christian Riekoff](#)
Allows Processing to communicate with control devices like joysticks and joypads.

» [MyGUI](#)

by [Markavian](#)
Implements buttons, radio button, checkboxes, sliders, and scroll bars.

» [Interfascia](#)

by [Brandon Berg](#)
Provides a toolkit of standard interface widgets like text fields, buttons, checkboxes, sliders, etc.

» [SpringGUI](#)

by [Philipp Seifried](#)

» [xmirepib](#)

by [Burak Arslan](#)
An XML-RPC library for Processing. XML-RPC is a widely adopted Remote Procedure Calling protocol that works over the Internet.

» [EnvironmentXML](#)

by [Uman Hagur](#)
Enables simple access to and from the EnvironmentXML website, which hosts a repository of data feeds from different environments around the world.

» [ID3](#)

by [Jorge Cardoso](#)
Allows Processing to read the ID3v1.1 tags from MP3 files.

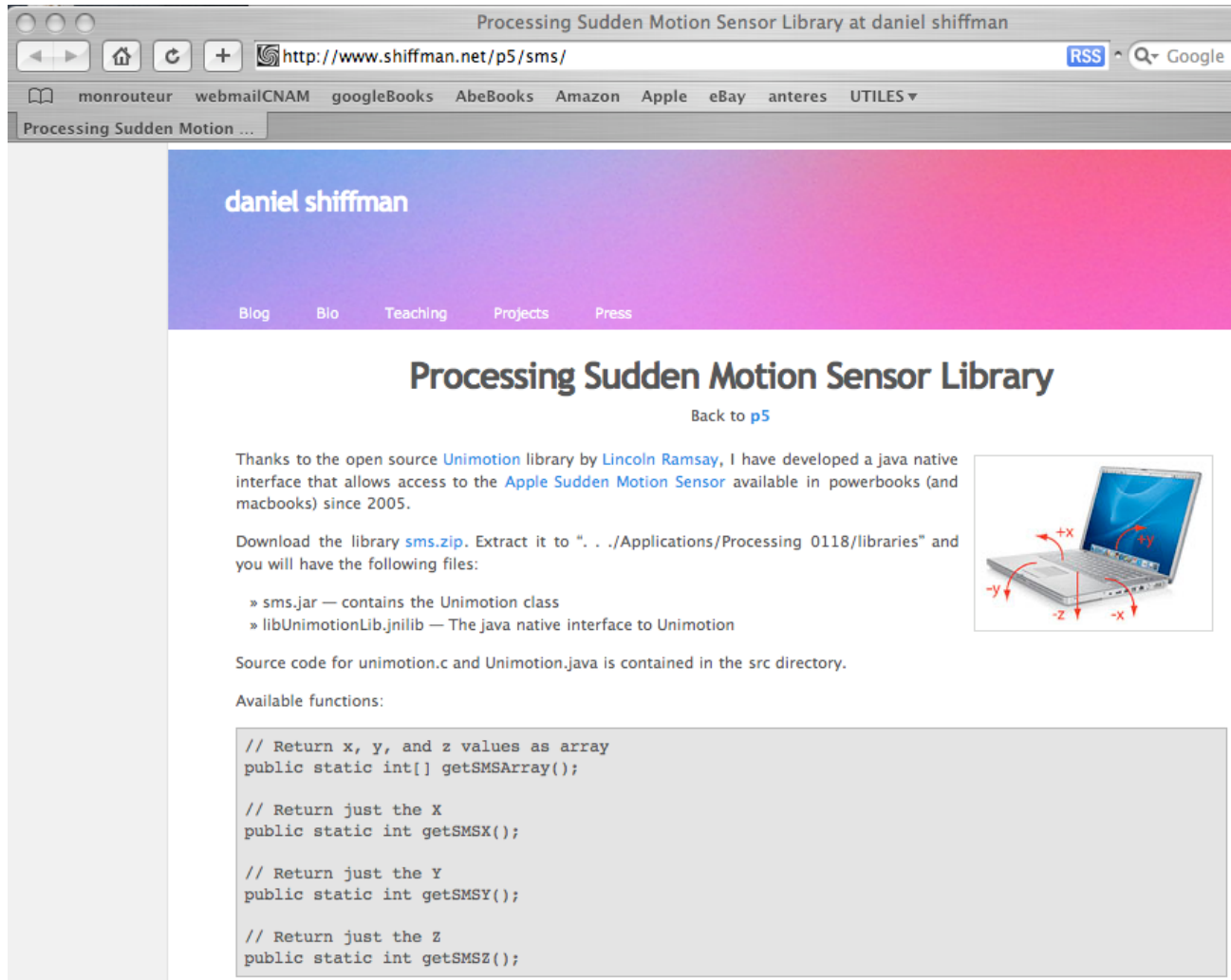
» [QRCode](#)

by [Daniel Shiffman](#)
Reads QR Code images, a two-dimensional barcode format.

» [SocialNetworks Library](#)

by [Todd Holloway](#)
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)



Processing Sudden Motion Sensor Library at daniel shiffman

<http://www.shiffman.net/p5/sms/> RSS Google

monrouteur webmailCNAM googleBooks AbeBooks Amazon Apple eBay anteres UTILES ▾

Processing Sudden Motion ...

daniel shiffman

Blog Bio Teaching Projects Press

Processing Sudden Motion Sensor Library

[Back to p5](#)

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.

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

Source code for unimotion.c and Unimotion.java is contained in the src directory.

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();

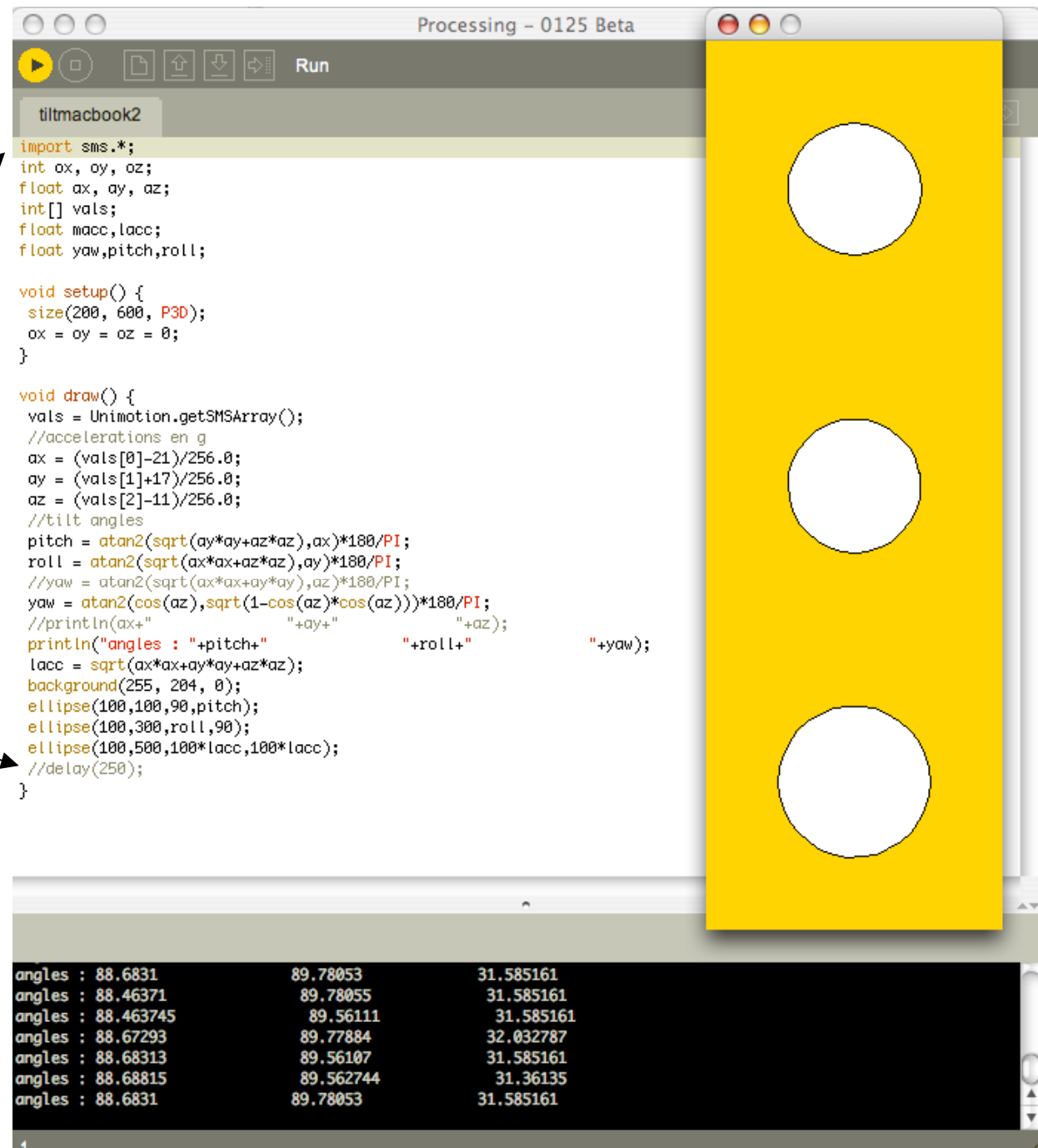
// Return just the Z
public static int getSMSZ();
```



Un petit
sketch de demo

import

Pb. du
framerate



```
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(100,100,90,pitch);
  ellipse(100,300,roll,90);
  ellipse(100,500,100*lacc,100*lacc);
  //delay(250);
}
```

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 bibliothèque sonore ESS

<http://www.tree-axis.com/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);

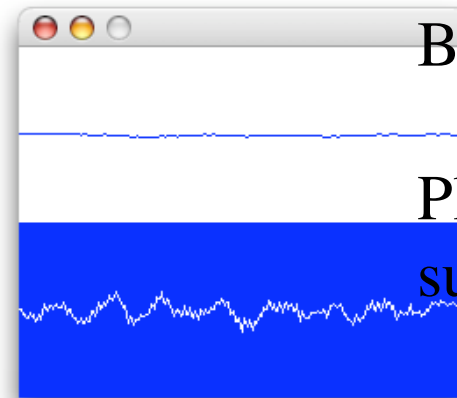
  // create a new AudioStream (4k buffer)
  myStream=new AudioStream(myInput.size);
  streamBuffer=new float[myInput.size];

  // 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);
```

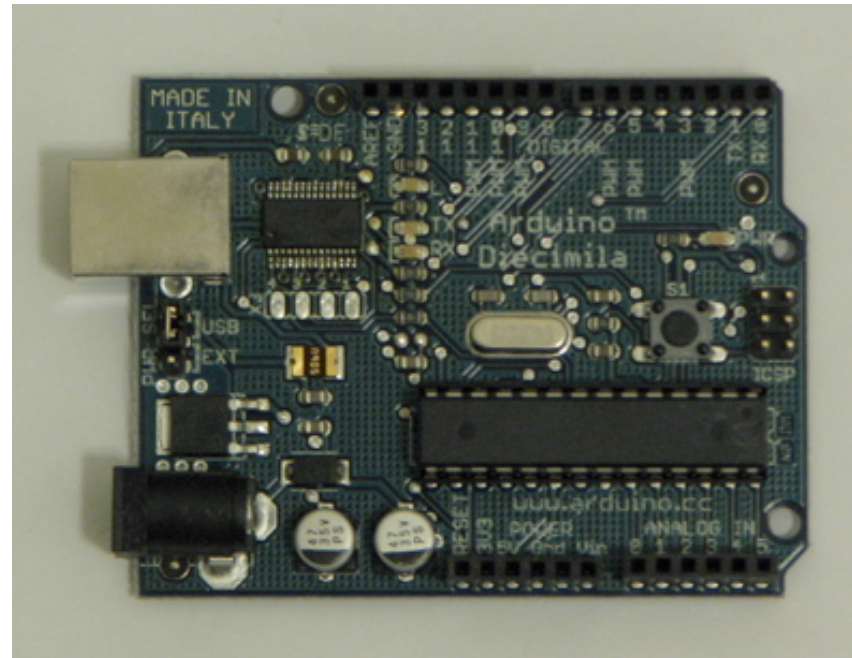


Basée sur JavaSound

Plusieurs très bons ex.
sur le site

Ici : squirrel-echo
(étudier ce code !)

Plus tard :



<http://www.arduino.cc>