M1 ENJMIN - MJV102

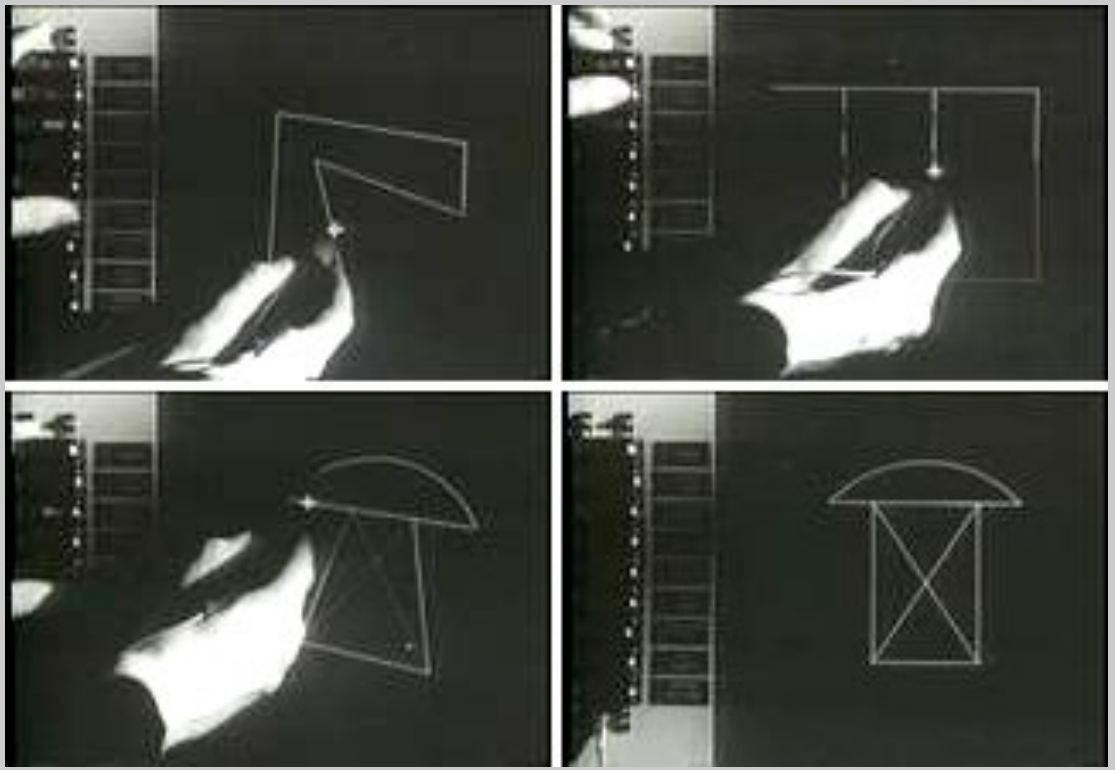
Conception et développement informatique

2 à 6 oct. 2017

Pierre Cubaud cubaud @ cnam.fr
Viviane Gal gal @ cnam.fr

le cnam
École nationale du jeu et des médias interactifs numériques enjmin

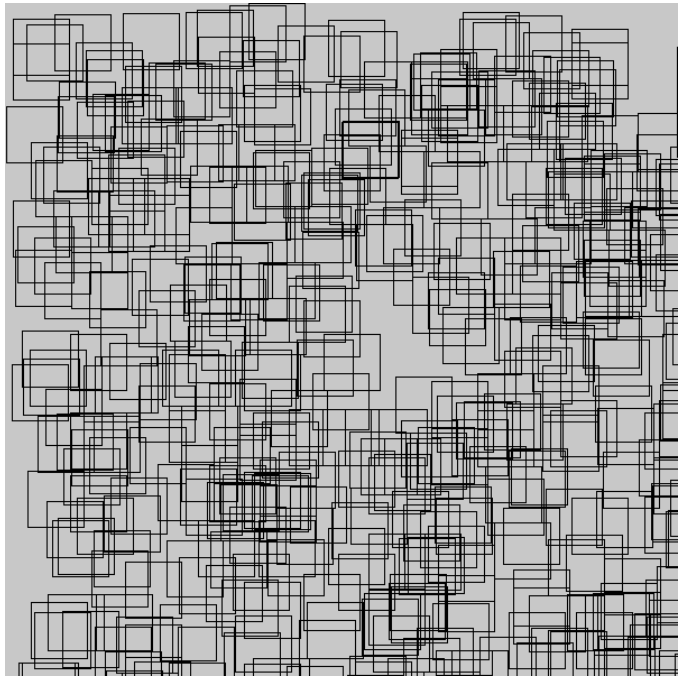
JOUR 3. L'interaction



Y. Sutherland
sketchpad

Processing s'anime

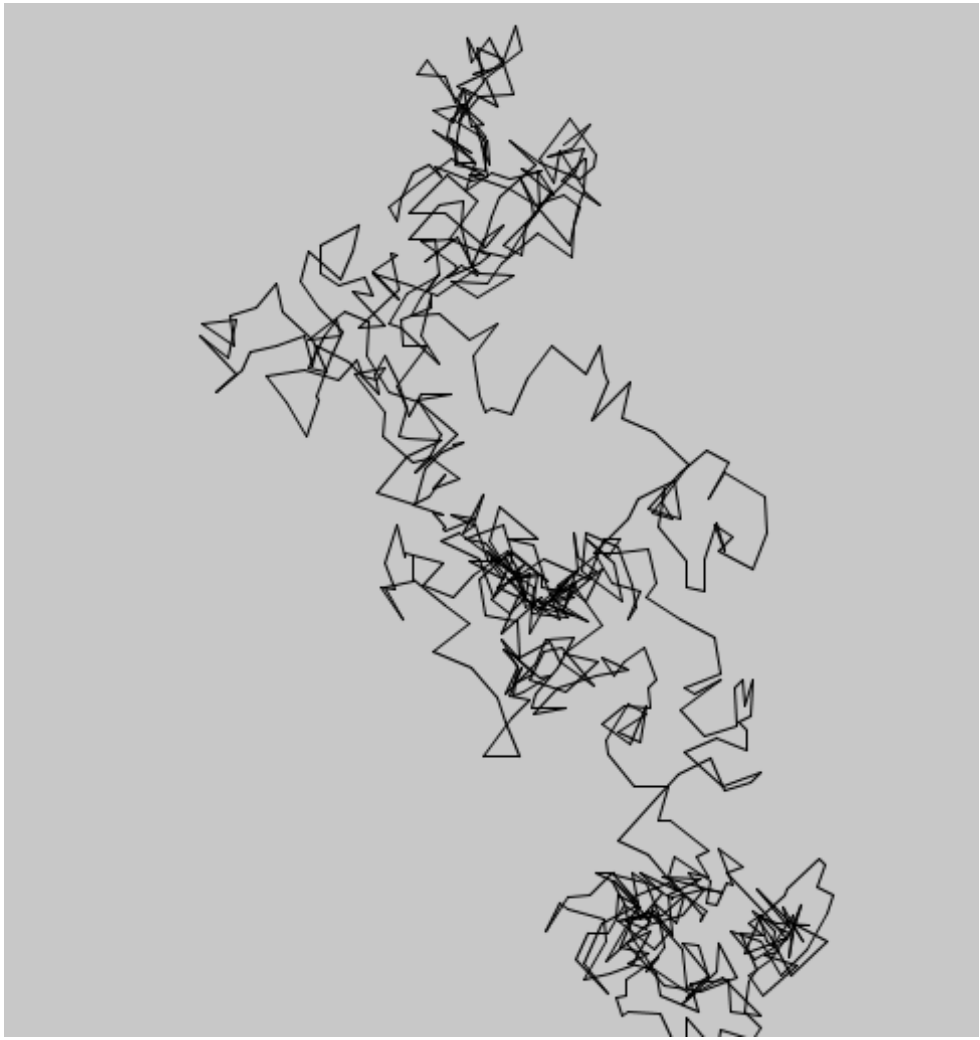
setup(){…} draw(){…}
frameRate(…) frameCount



```
void setup() {
  size(600,600);
  noFill();stroke(0);
  background(200);
}

void draw() {
  float x=random(0,width);
  float y=random(0,height);
  rect(x,y,50,50);
}
```

(modereactif1.pde)

Mouvement brownien



```
float ox=300;
float oy=300;
float x,y;

void setup() {
  size(600,600);
  stroke(0);
  smooth();
}

void draw() {
  x = ox + random(-20,+20);
  y = oy + random(-20,+20);
  line(ox,oy,x,y);
  ox = x;
  oy = y;
}
```

(brownien1.pde)

Variante : particule brownienne

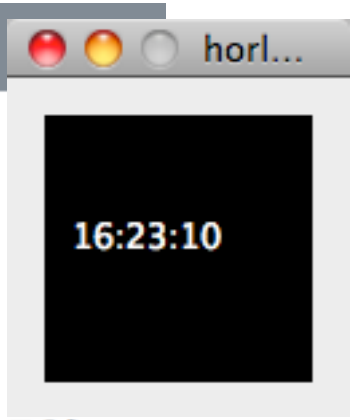
animation2

```
float x=300;
float y=300;

void setup() {
  size(600,600);
  fill(100);noStroke();
  smooth();
}

void draw() {
  background(200);
  x = x + random(-2,+2);
  y = y + random(-2,+2);
  ellipse(x,y,50,50);
}
```

Une horloge

horloge

```
void draw() {
  background(0);
  int s = second();
  int m = minute();
  int h = hour();
  text(str(h)+":"+str(m)+":"+nf(s,2),10,50);
}
```



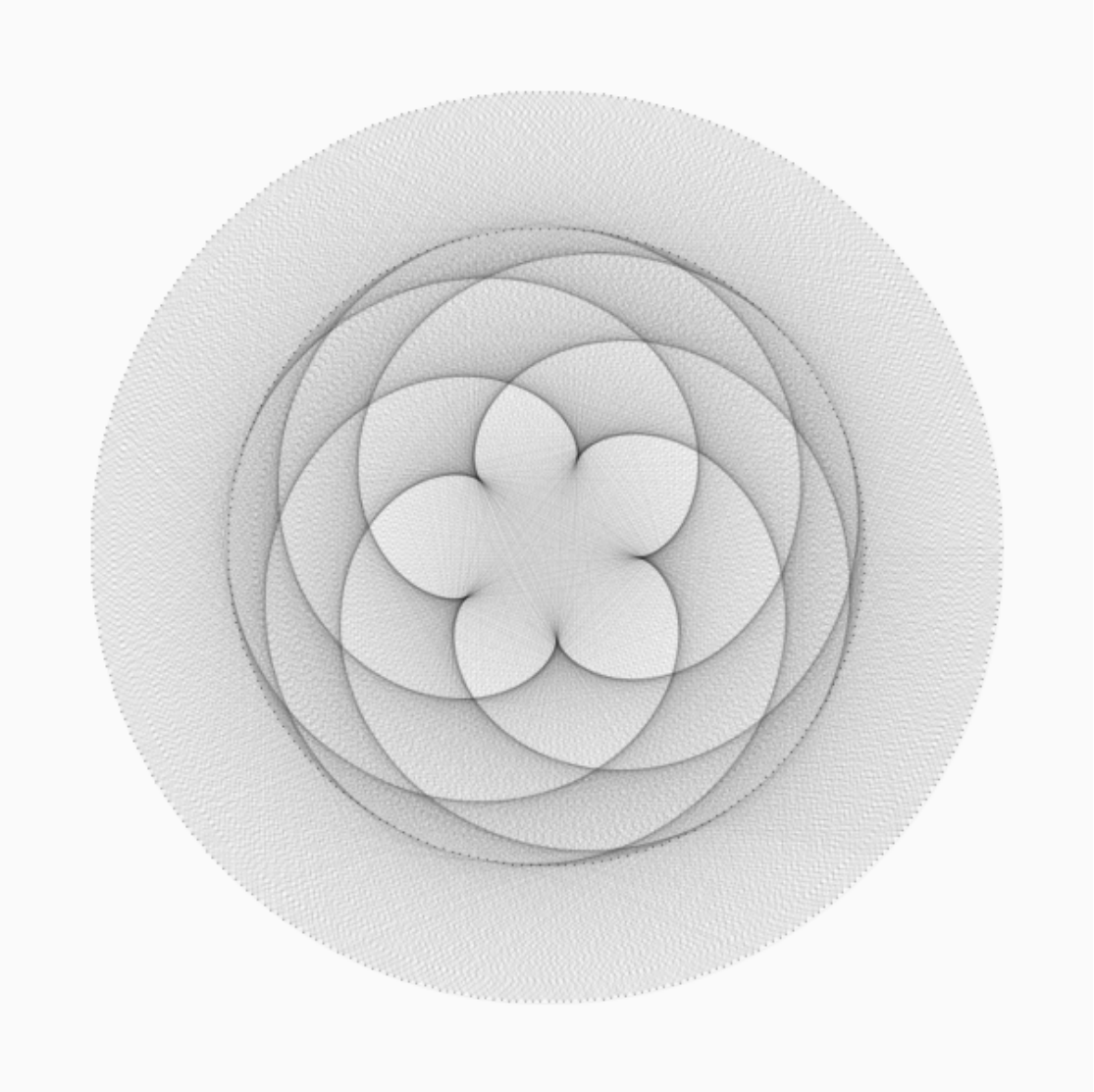
nf() = ???

horlogeMilli

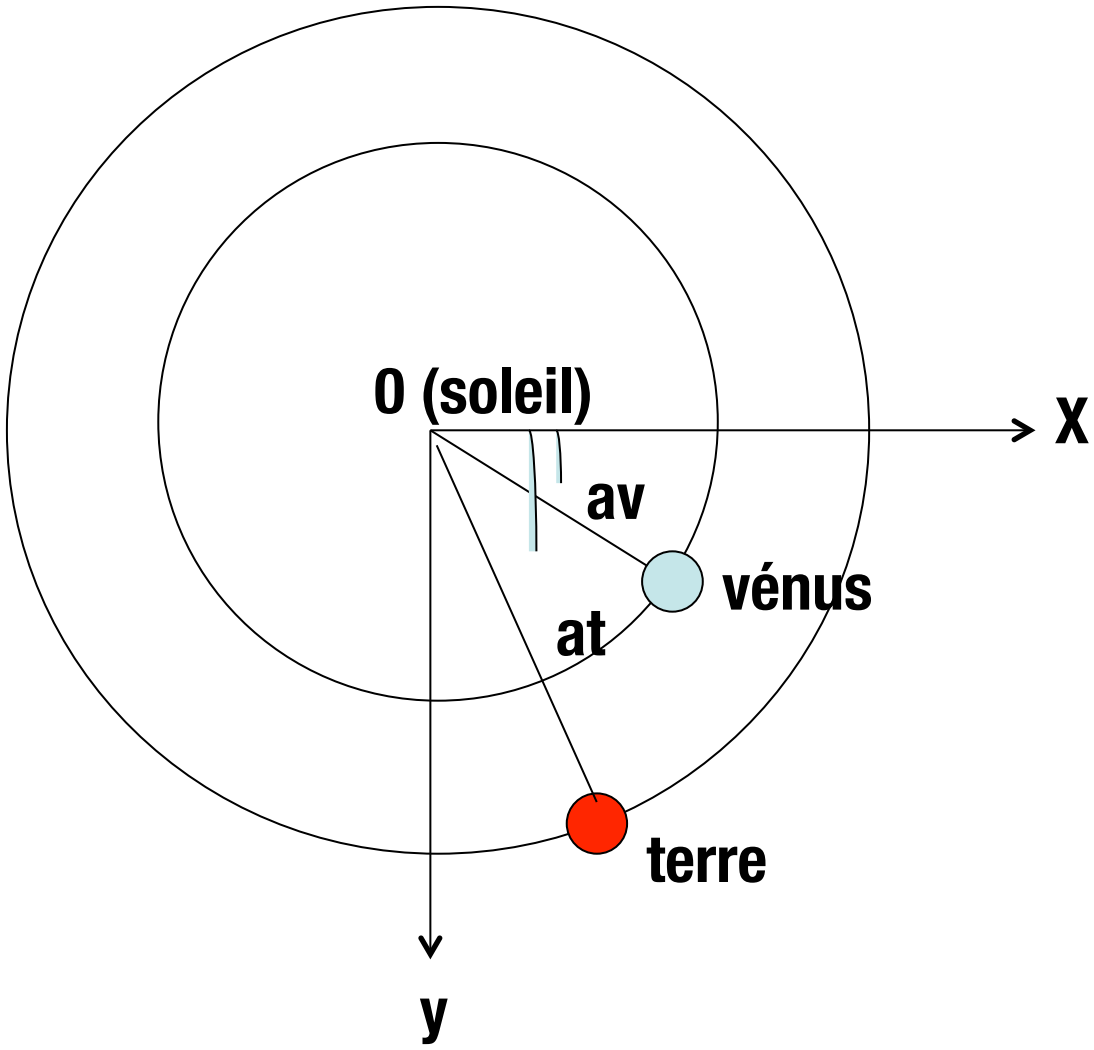
```
void setup() {
  frameRate(10);
}

void draw() {
  background(0);
  text(millis(),10,50);
}
```

essayer plusieurs variantes

Orbites planétaires



Un peu de trigonométrie



d(soleil, venus) = 0.7 UA
=>175 pixels
d(soleil, terre) = 1 UA
=> 250 pixels

revolution terre = 365j
revolution vénus = 225j

xv = 175*cos(av)
yv = 175*sin(av)

av += 2π/225

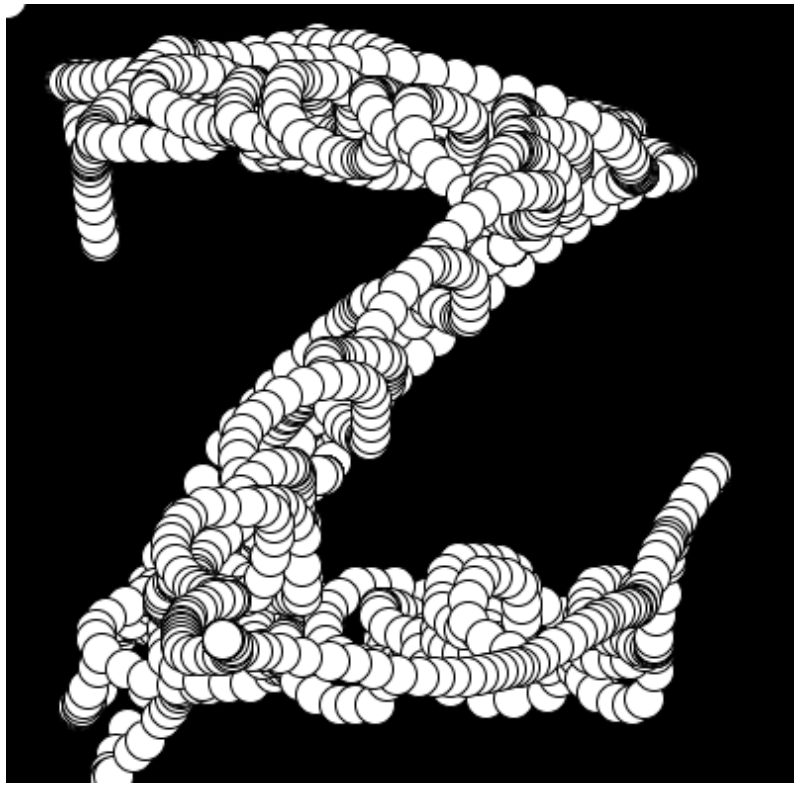
```
float at,av,dt,dv;

void setup(){
  size(600,600);
  smooth();
  background(250);
  //stroke(150,0,250,10);
  stroke(0,10);
  ellipseMode(CENTER);
  noFill();
  ellipse(300,300,500,500);
  ellipse(300,300,350,350);
  dt = 2*PI/365/1;
  dv = 2*PI/225/1;
  at = av = 0;
}
```

```
void draw(){
  // position terre
  float xt = 250*cos(at);
  float yt = 250*sin(at);
  //ellipse(300+xt,300+yt,10,10);
  // position venus
  float xv = 175*cos(av);
  float yv = 175*sin(av);
  //ellipse(300+xv,300+yv,10,10);
  // arc
  line(300+xt,300+yt,300+xv,300+yv);
  // avance des planetes
  at = (at+dt)%(2*PI);
  av = (av+dv)%(2*PI);
}

void keyPressed(){
  saveFrame("rosace###.png");
}
```

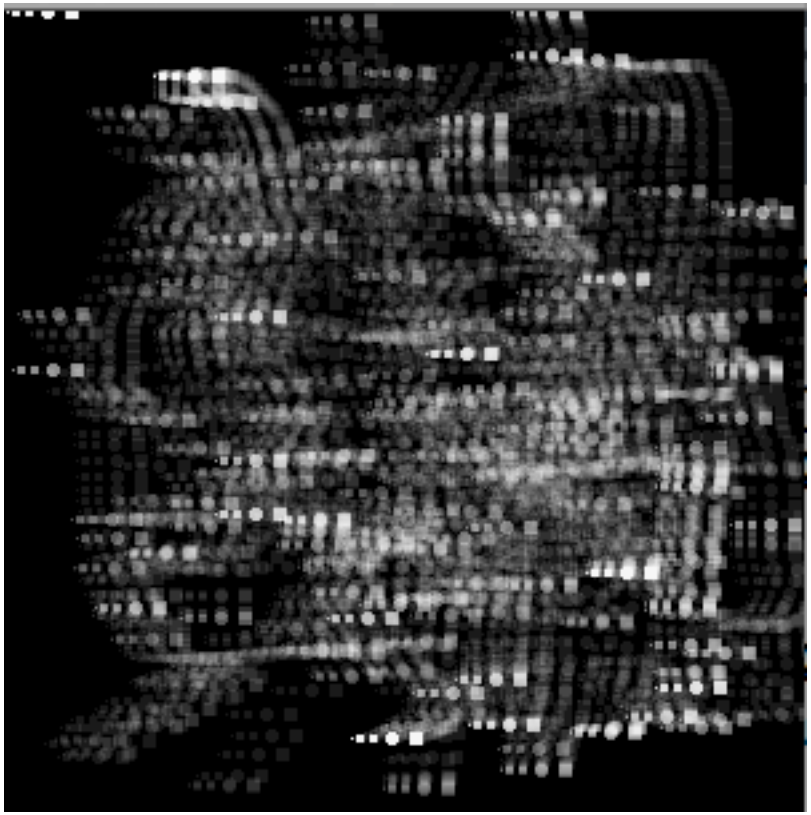
(planetes.pde)

Souris, curseur

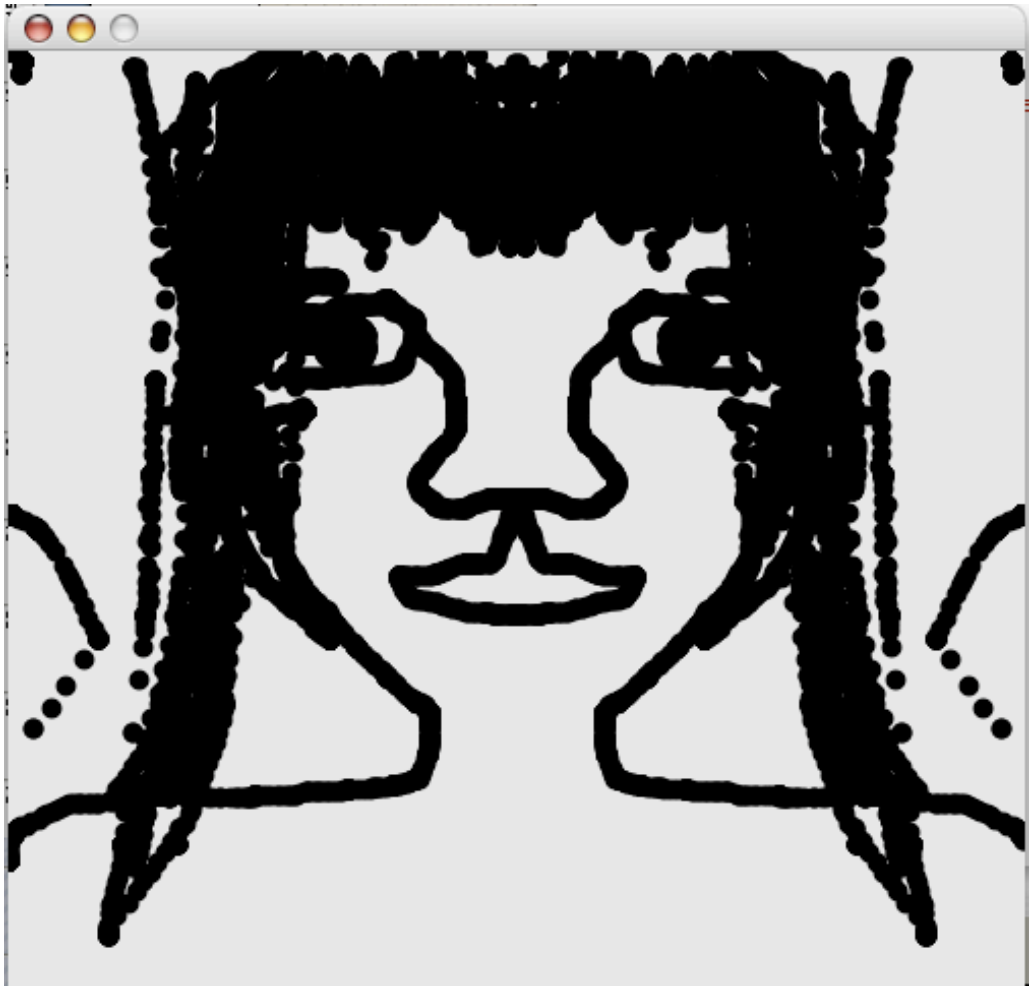
```
void setup(){
  size(400,400);
  smooth();
  background(0);
  noCursor();
}

void draw(){
//  background(0);
  int x = mouseX;
  int y = mouseY;
  ellipse(x,y,20,20);
}
```



(demosouris.pde)



Symétries



```
boolean dessin = false;

void setup() {
  size(500, 500);
  background(226);
  smooth();
  strokeWeight(20); stroke(0, 100);
}

void draw() {
  point(mouseX,mouseY);
  point(width-mouseX, mouseY);

  //// variante pour symetrie centrale
  //point(mouseX,height-mouseY);
  //point(width-mouseX,height-mouseY);

}
```

(symetries.pde)

Gestion d'évènements

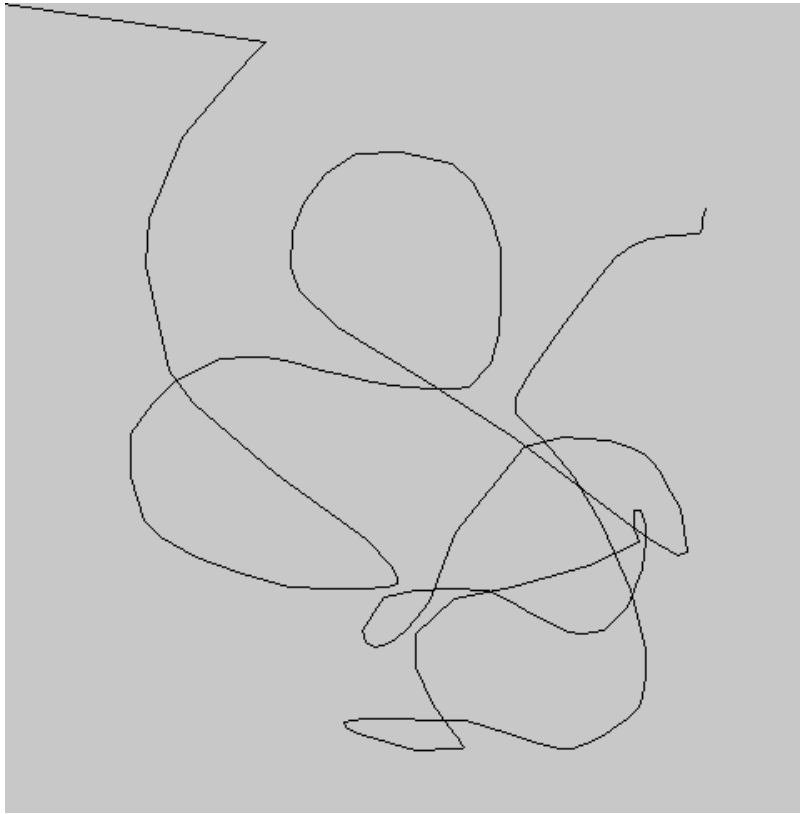
Pour que le code réagisse aux actions de l'utilisateur, il faut renseigner les fonctions appropriées :

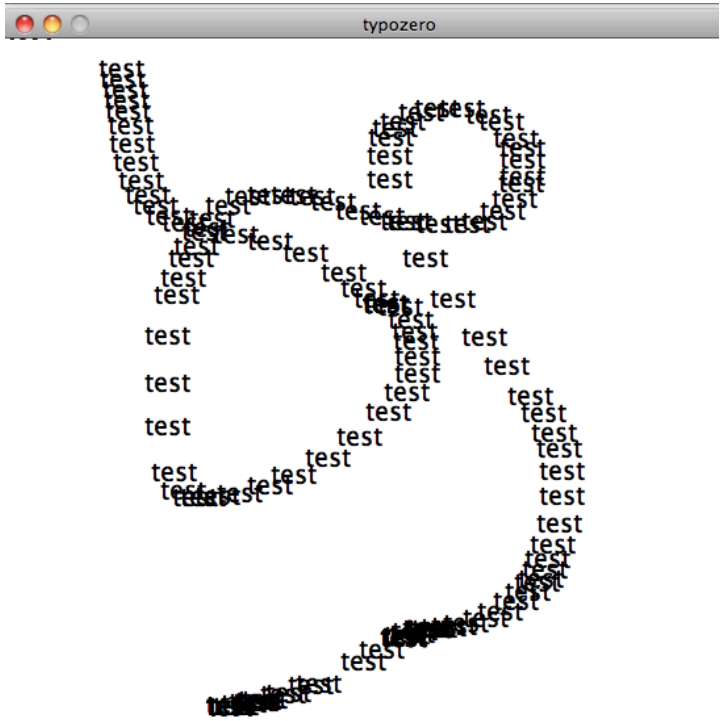
void mousePressed()
void mouseReleased()
void mouseMoved()
void mouseDragged()
+ utiliser les variables pré-définies mouseX mouseY

void keyPressed()
void keyReleased()
+ utiliser la variable pré-définie key

linepmouse

```
void setup(){
  size(600,600);
}
void draw(){
  line(pmouseX,pmouseY,mouseX,mouseY);
}
```




typozero
test

une (mauvaise) machine à écrire

machineecrire

```
String letexte="";

void setup(){
  size(800,600);
  stroke(255);fill(255);
}

void draw(){
  background(0);
  text(letexte,10,10);
}

void keyPressed() {
 letexte += key;
}
```

pas de retour (feedback) utilisateur !

slider sur particule

Les images avec Processing

```
Pimage img = loadImage("gnagna.jpg");
image(img, 30,40, 640, 480);
```

Coloriage de l'image : tint(…) noTint()

Accès aux pixels de l'écran : get(…) set(…) copy(…)
Pour une image : ima.get(…) ima.set(…)

filter(…) blend(…) mask(…)

loadPixels(…) Pixel[] updatePixels()

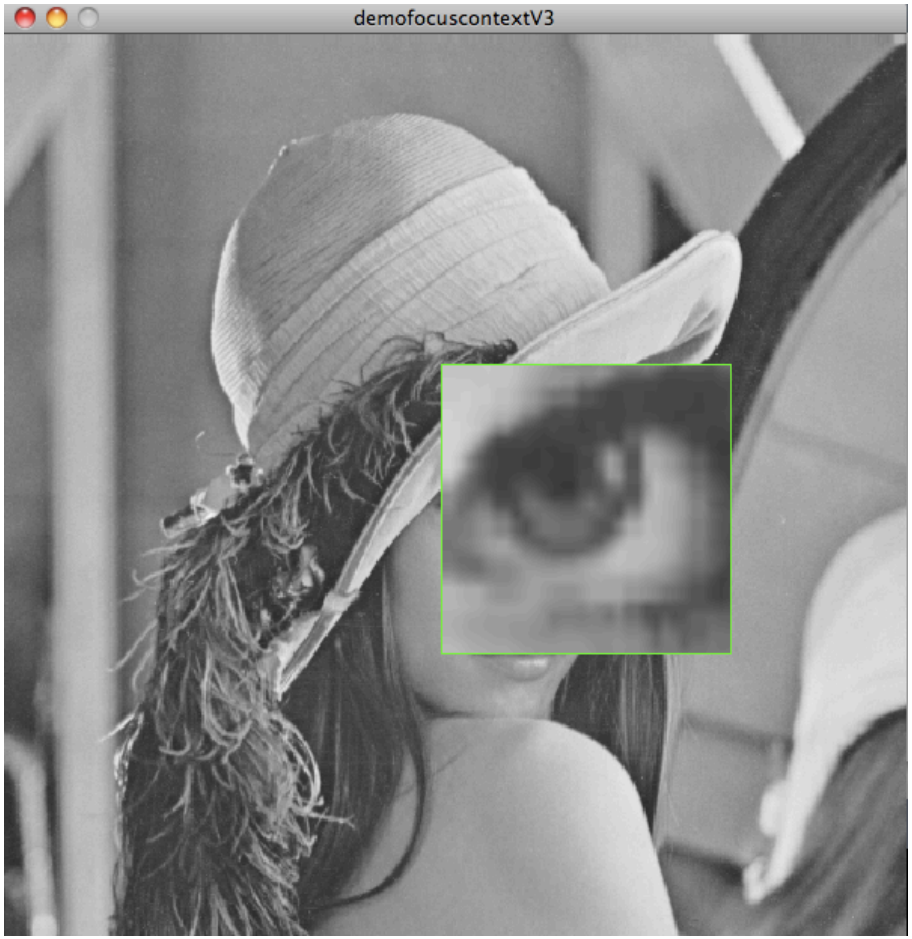
Hommage à Akaklikè

Processing - 0125 Beta

akaklike

```
PImage ima;

void setup(){
  ima=loadImage("Photo.jpg");
  size(ima.width,ima.height);
}

void draw(){
  copy(ima,mouseX,mouseY,30,30,mouseX,mouseY,30,30);
}
```

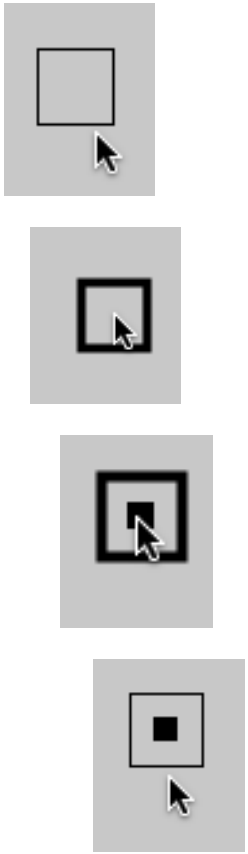
Loupe



```
PImage pgmat;
float ar;

void setup(){
 pgmat = loadImage("lenna.gif");
 ar = pgmat.width/float(pgmat.height);
 size(int(600*ar), 600);
}

void draw(){
  int x = mouseX;
  int y = mouseY;
  image(pgmat, 0,0, width,height);
  copy(x-16,y-16,32,32, x-96, y-96, 192,192);
  stroke(0,255,0);noFill();rect(x-96, y-96, 192,192);
}
```

(demofocuscontextV3.pde)

Dessine-moi un bouton (radio)



état 1 : non sélectionné, non désigné

entrée de zone : "roll over"

état 2 : non sélectionné, désigné

clic

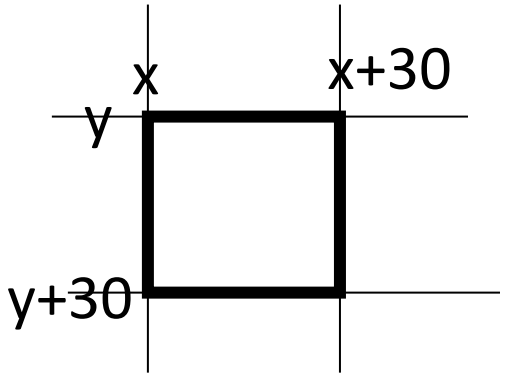
état 3 : sélectionné, désigné

sortie de zone

état 4 : non sélectionné, désigné

Détection du rollover :

bouton = un carré
en coord (x,y) de largeur
30 pixels




```
si    (mouseX > x) et (mouseX < x+30)
      et (mouseY >y) et (mouseY < y+30)
alors le curseur est dans la boite du bouton
sinon il est dehors
```

⇒une variable booléenne ("boolean")
pour le rollover + une autre pour la selection

début du code

monboutonV0

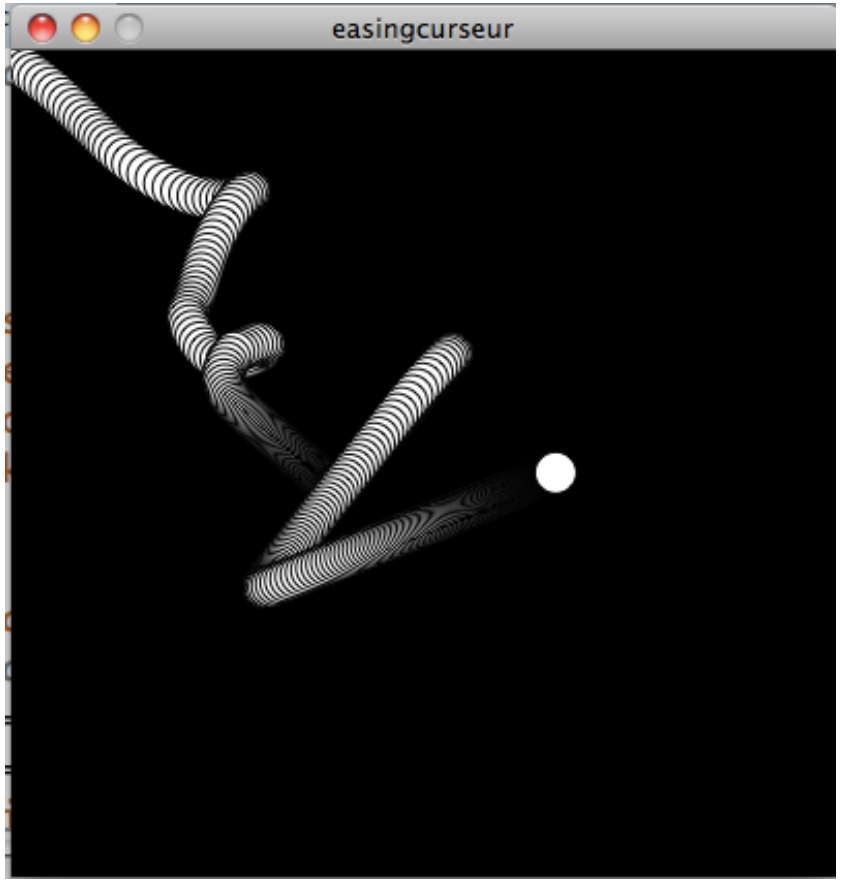
```
int x,y;
boolean rollover, selected;

void setup() {
  size(200,200);
  x = 50; y = 50;
  rollover = false; selected = false;
}

void draw() {
  background(200);
  stroke(0);noFill();
  if (rollover) strokeWeight(4); else strokeWeight(1);
  rect(x,y,30,30);
  if (selected) {
    noStroke();fill(0);
    rect(x+10,y+10,10,10);
  }
}
```

Suite du code :

```
void mouseMoved() {
 int mx = mouseX;
 int my = mouseY;
 if (mx > x && mx < x + 30 && my > y && my < y + 30)
   rollover = true;
 else
   rollover = false;
}

void mousePressed() {
 if (rollover)
   selected = ! selected;
}
```

essayer d'autres
formes de boutons
et d'autres feedbacks

Easing

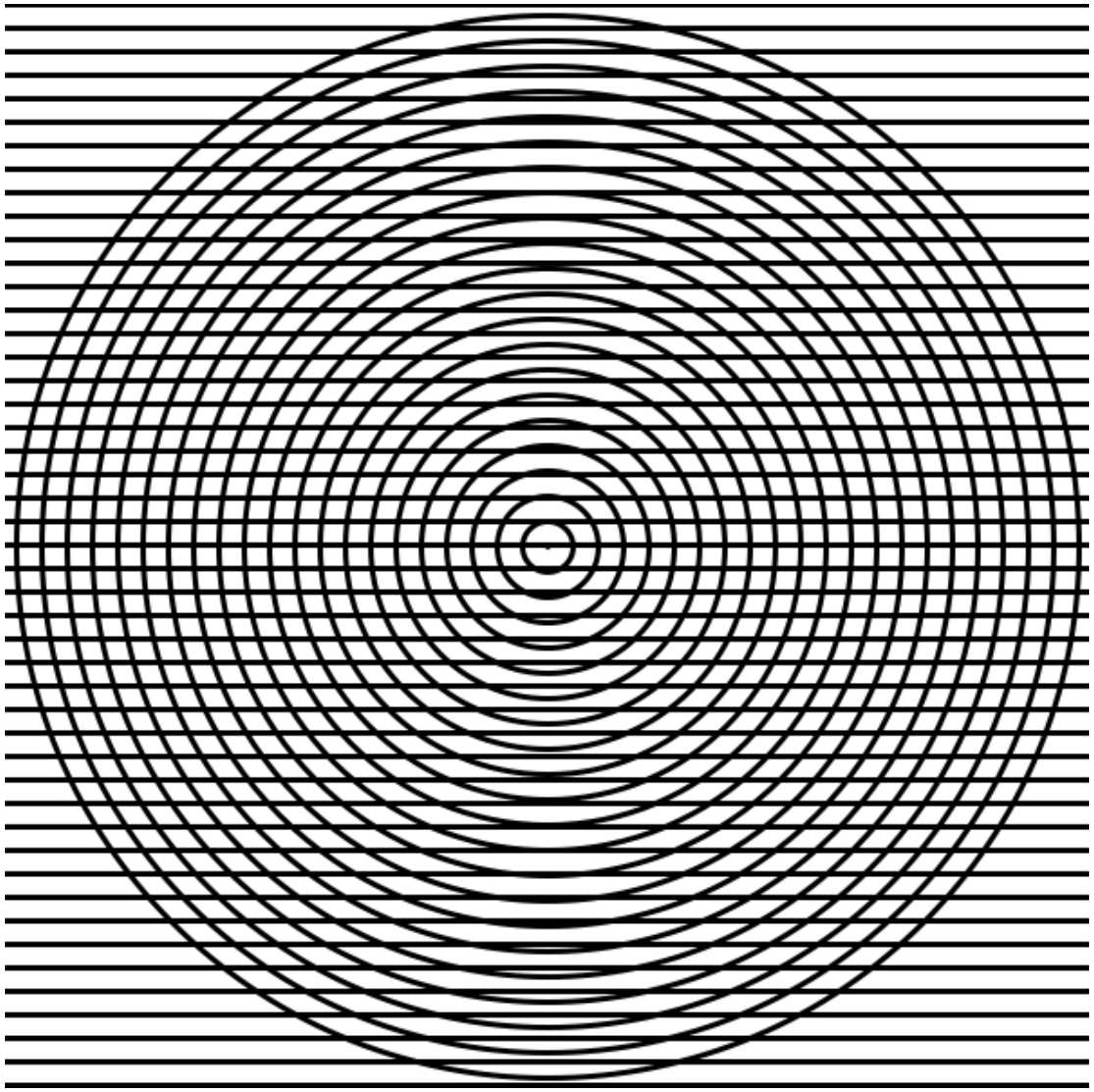


easingcurseur

```
// d'apres Geridan & Lafargue "Processing"
float x=0;
float y=0;
float ease=0.1;

void setup(){
  size(400,400);
  smooth();
  background(0);
}

void draw(){
  background(0);
  x += (mouseX -x)*ease;
  y += (mouseY -y)*ease;
  ellipse(x,y,20,20);
}
```

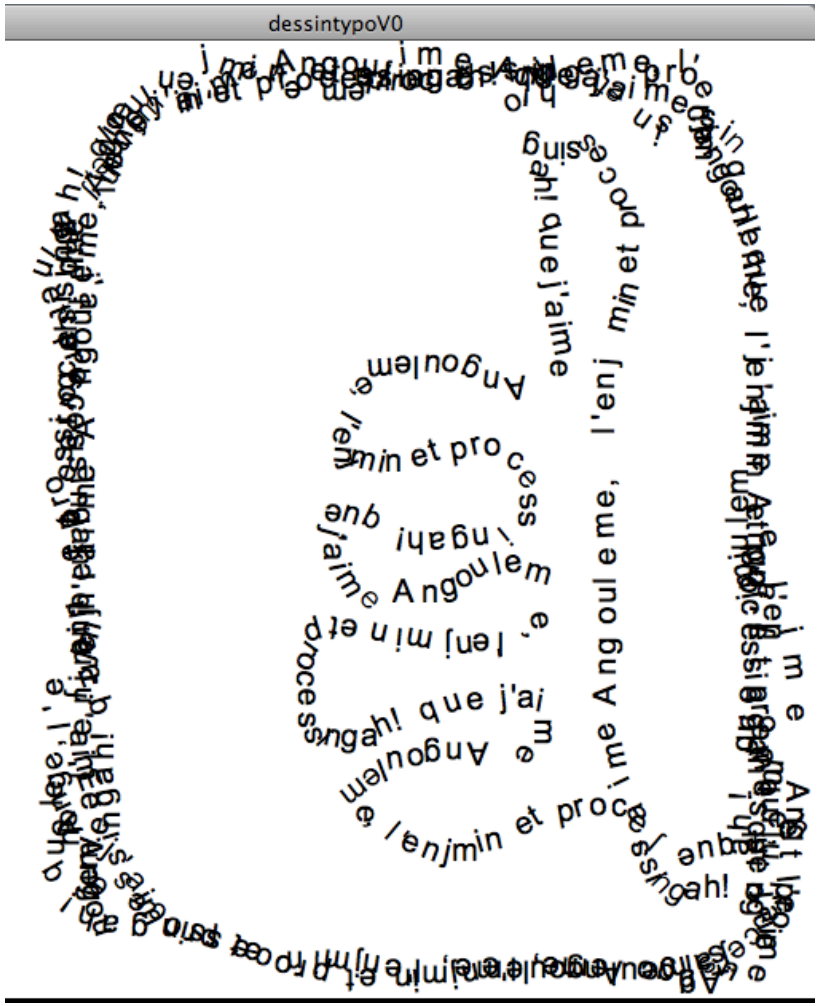
plusieurs autres
méthodes (fonction sinus etc)

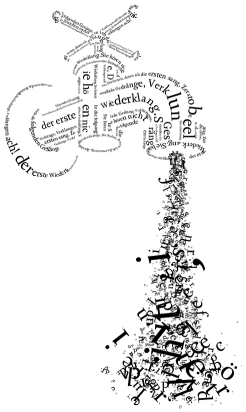
Le moiré interactif, avec la souris



```
void draw(){
  background(255);
  //dessin grille
  int y = 0;
  while (y<height){
    line(0,y,width,y);
    y += delta;
  }
  //dessin cibles
  float delta2=map(mouseY,0,height,delta,2*delta);
  pushMatrix();
  translate(mouseX, mouseY);
  int r = 0;
  while (r<height/2){
   ellipse(0,0,2*r,2*r);
   r += delta2+1; //ou delta ou etc.
  }
  popMatrix();
}
```

(moiresouris.pde)





d'après programme P.2.3.3 de "Design génératif"

```
float x, y, ox, oy, d;
float stepSize;
PFont font;
String letters = "ah! que j'aime Angouleme, l'enjmin et processing";
int counter = 0;

void setup() {
  size(600,600);
  background(255);
  fill(0);
  smooth();
  cursor(CROSS);
  font = createFont("Arial",10);
  textFont(font,20);
  textAlign(LEFT);
  stepSize = 0;
  x = mouseX;
  y = mouseY;
  ox = x;
  oy = y;
}
```

```
void draw() {
  x = mouseX;
  y = mouseY;
  d = dist(x,y, ox,oy);

  if (d >= stepSize) {
   float angle = atan2(y-oy, x-ox);
   pushMatrix();
   translate(x, y);
   rotate(angle);
   //textFont(font,fontSizeMin+d);
   char newLetter = letters.charAt(counter);
   text(newLetter, 0, 0);
   counter = (counter+1)%letters.length();
   popMatrix();

   stepSize = textWidth(newLetter);
   ox = x;
   oy = y;
  }
}
```

(dessintypoV0.pde)

spline