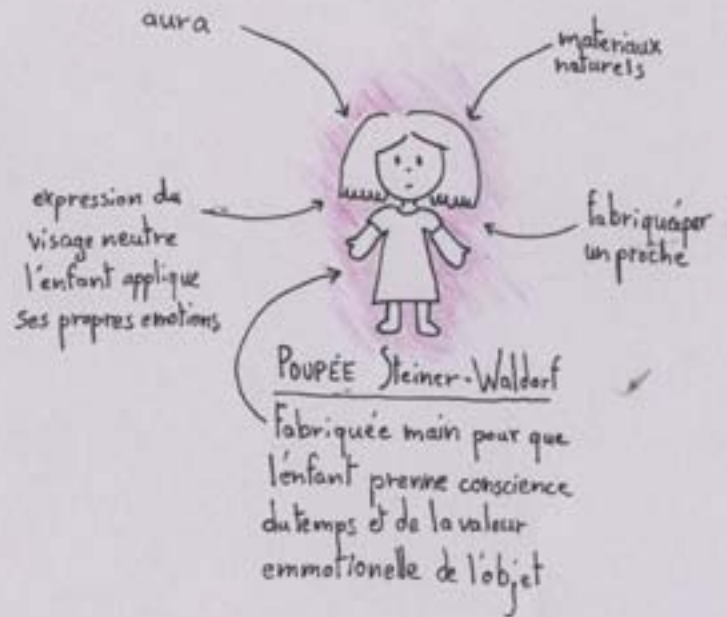
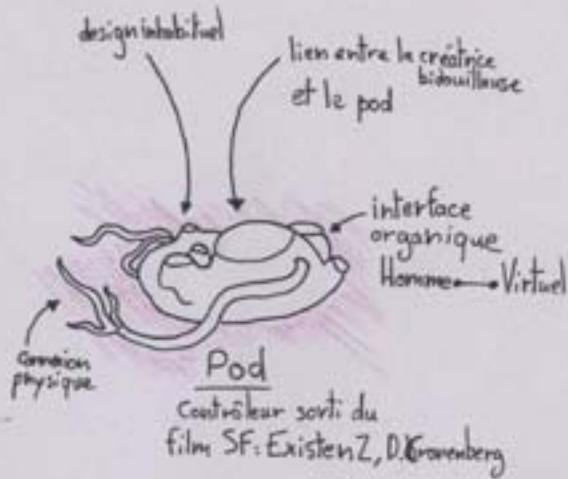


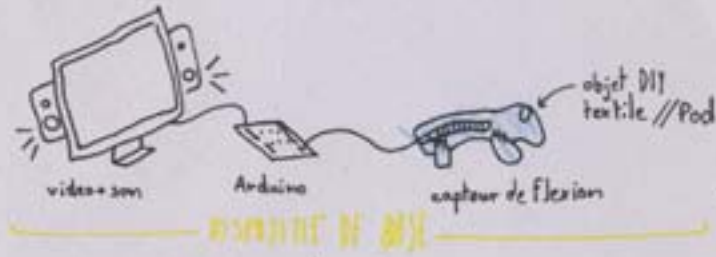
# DÉSORDRE MUSIGRAPHIQUE



# PROJET ARDUINO



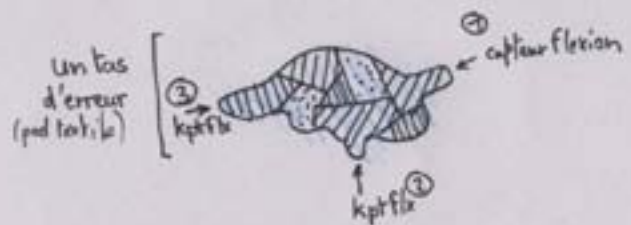
**ABANDONNE**  
charge émotionnelle de l'objet DIY



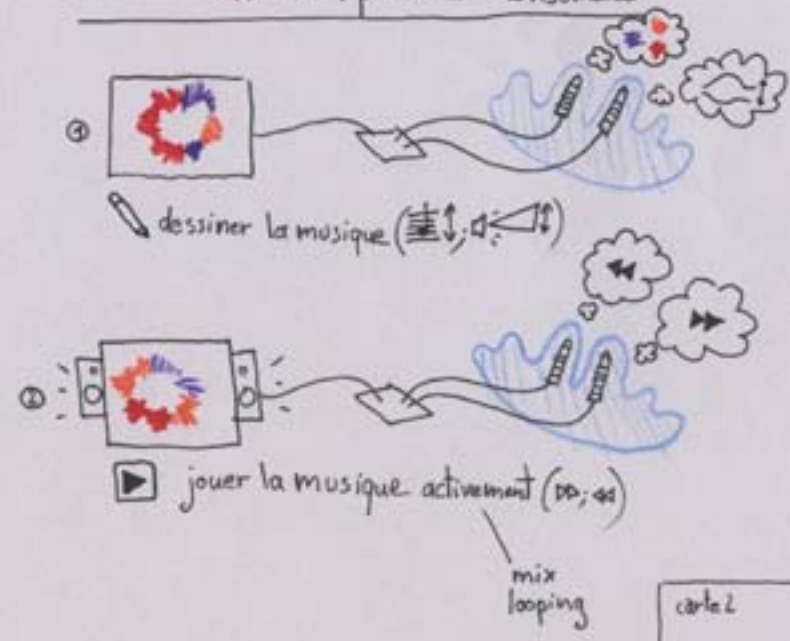
**ABANDONNE**  
apprendre de l'erreur

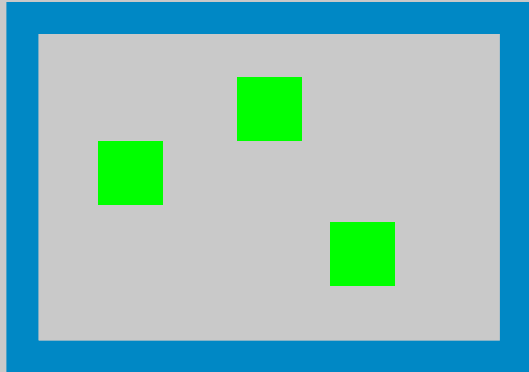
**RETURN**  
devenir R. producteur-consomateur

- Si pas de flexion  $x < a$   
Alors jouer vidéo souvenir heureux
- Si flexion moyenne  $a < x < b$   
Alors jouer vidéo souvenir triste dur personnel
- Si flexion forte  $b < x$   
Alors jouer vidéo souvenir triste dur personnel proche

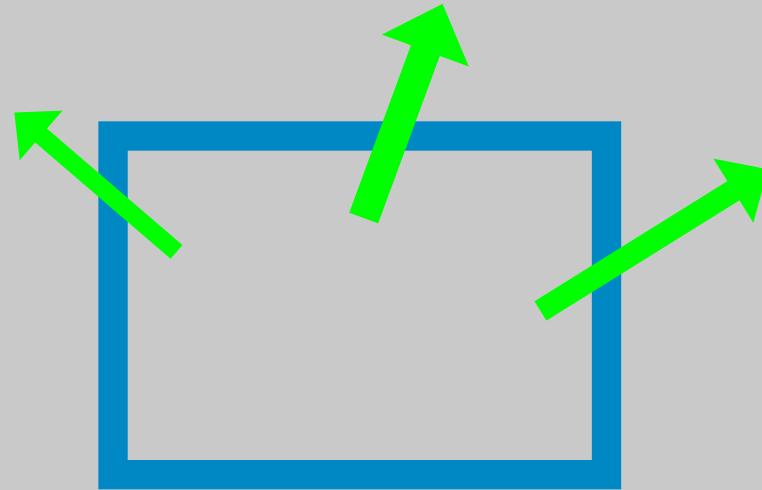


- Si ①②③ pas de flexion Alors Glitch
- Si ①② pas de flexion Alors Glitch + Son
- Si ① pas de flexion Alors video+Glitch+Son
- Si flexion ① Alors video+Son





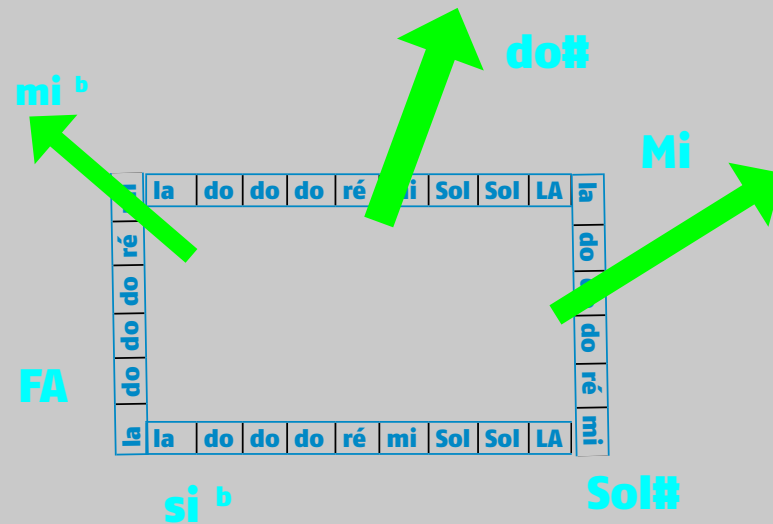
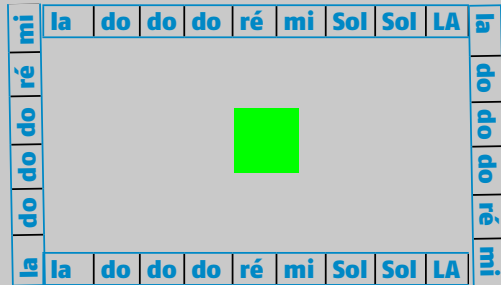
**En graphisme, concevoir un espace d'expression pour autrui revient à lui placer des limites, des restrictions. On entre dans l'univers du kit ou de la personnalisation standardisée.**



### **OBJECTIFS :**

- **mettre en évidence ces limites**
- **créer des passages de transgression des règles**

# DÉSORDRE MUSIGRAPHIQUE



**mettre en évidence la limite**

**je met en place un parallèle entre  
graphisme et musique.**

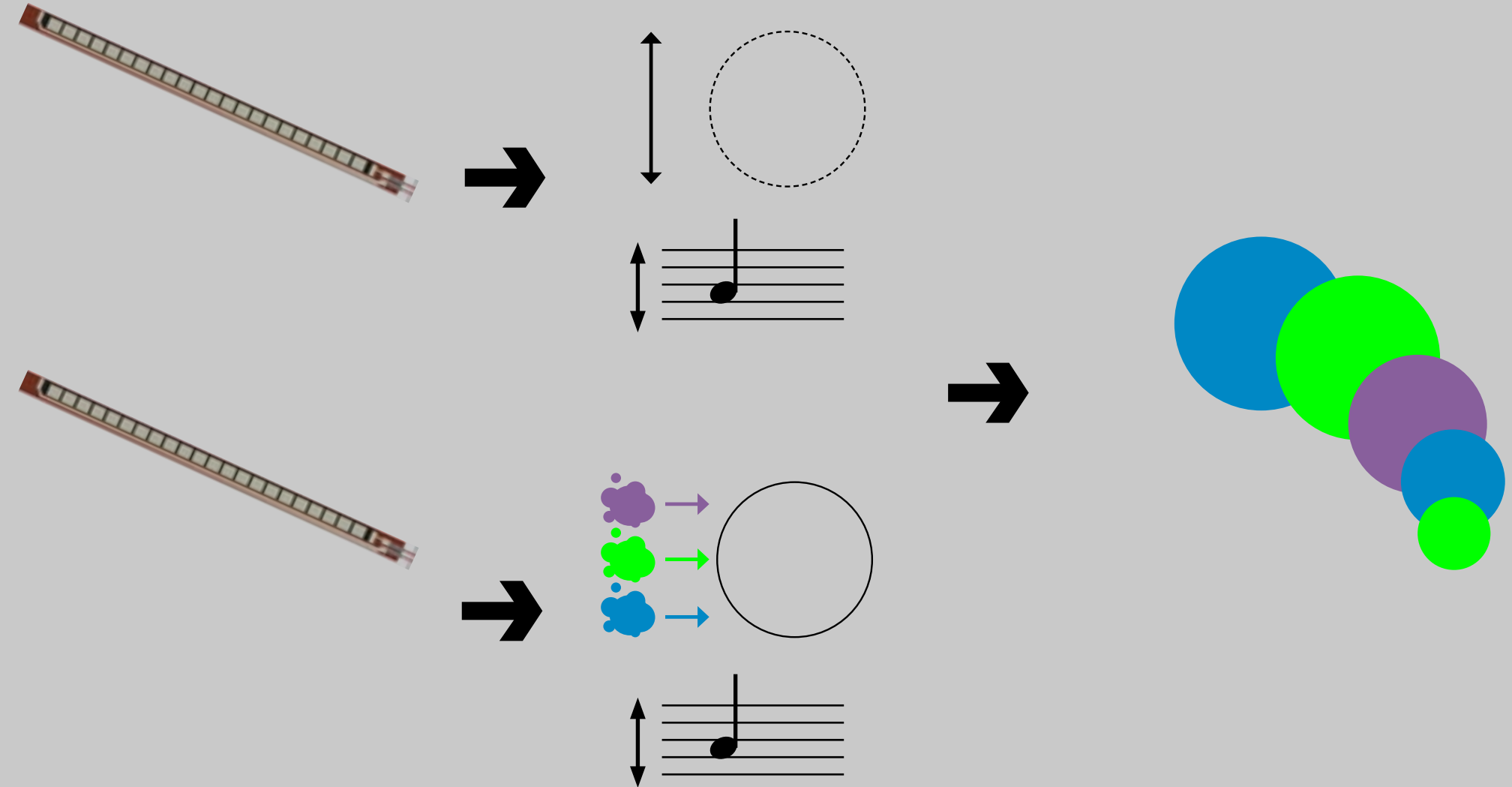
**J'oblige l'utilisateur à naviguer entre 5  
notes de hauteurs différentes, harmo-  
nisées. Dans ce fonctionnement**

**créer des passages de transgression**

**je donne la possibilité à l'utilisateur d'activer  
un système différent qui lui permet d'ouvrir son  
champ de possibilité.**

**Il est alors possible de bousculer l'harmonie par  
une gamme complète.**

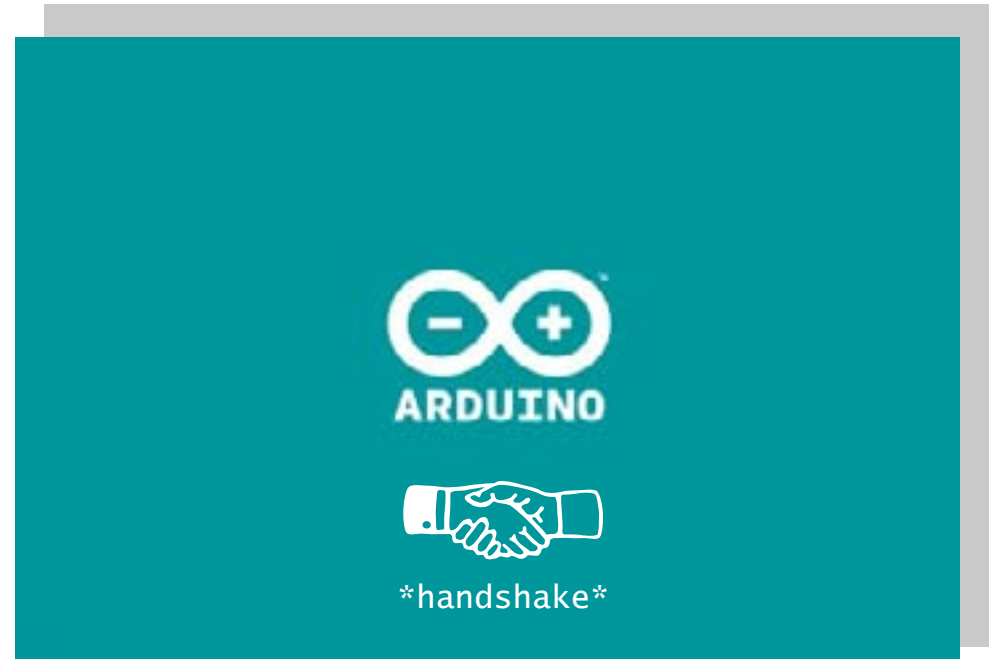
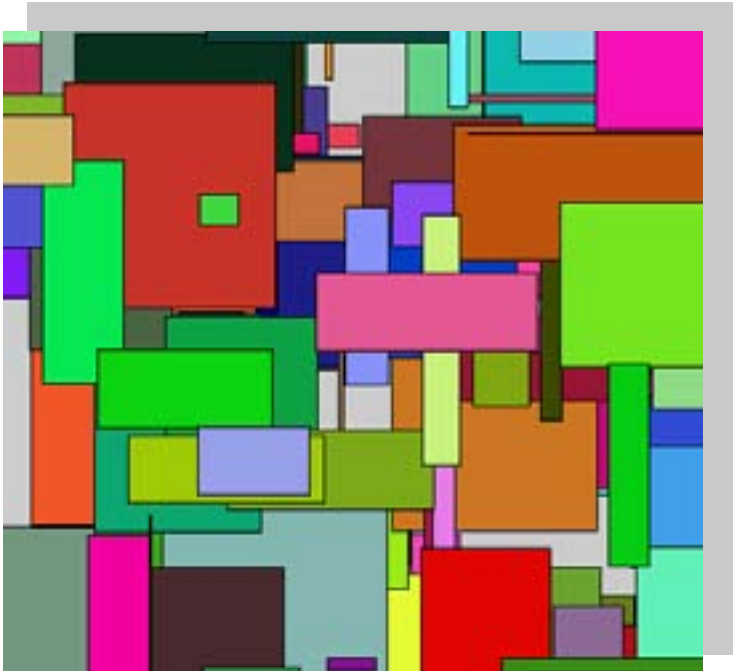
# PRINCIPE DE BASE



## 2 capteurs de flexions

utilisation de capteur de pression à la manière de joystick pour une expérience gestuelle différente de la pratique d'instrument de musique conventionnel. Pratique intuitive

## Génération de forme et de son



## Unlimited Art - Generating multi-part music while drawing

Tutoriel SoundCipher, qui m'a permis d'aborder cette librairie afin obtenir des superpositions mélodiques et des harmonies.

[www.explodingart.com/soundcipher/tutes](http://www.explodingart.com/soundcipher/tutes)

## Serial Call response

Serial Call response est un Protocole Handshaking : processus automatisé de négociation qui établit les paramètres d'une communication entre deux entités avant que la communication commence. J'ai besoin d'un handshake entre la carte arduino et l'ordinateur.

[www.arduino.cc/en/Tutorial/SerialCallResponse](http://www.arduino.cc/en/Tutorial/SerialCallResponse)

# MINIM

**gestion de fichiers sons**

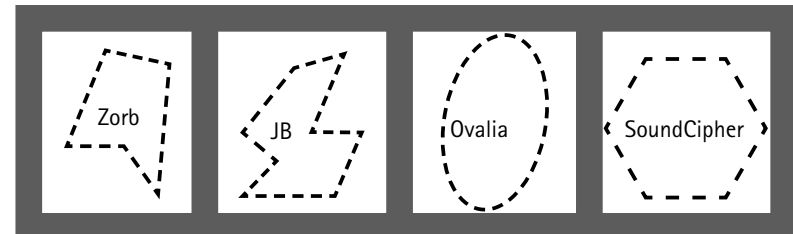


**musique algorithmique  
musique générative**

## **protocole MIDI**

protocole de communication et de commande permettant l'échange de données entre instruments de musique électronique

Une librairie contient une collection d'objet



# PROGRAMMATION

## Instanciation & attribution

**classe**



instanciation  
création d' occurrences  
de la classe SoundCipher

**instances**

attribution de la fonction .instrument()



**(11)**      **(88)**      **(98)**

correspondance sonorité d'instruments



vibraphone



fantasia



crystal

extrait du tableau de référence SC  
pour trouver les équivalences instrument  
<http://explodingart.com/soundcipher/reference.html>

```
import arb.soundcipher.*;
```

```
SoundCipher sc = new SoundCipher(this);  
SoundCipher sc2 = new SoundCipher(this);  
SoundCipher sc3 = new SoundCipher(this);
```

```
void setup()  
{  
  sc.instrument(11);  
  sc2.instrument(88);  
  sc3.instrument(98);  
}
```

### arb.soundcipher.constants.ProgramChanges

public static final float	<u>ALTO</u>	65.0f
public static final float	<u>ALTO SAX</u>	65.0f
public static final float	<u>ALTO SAXOPHONE</u>	65.0f
public static final float	<u>APPLAUSE</u>	126.0f
public static final float	<u>ATMOSPHERE</u>	99.0f



# Fonction musicale

création de la mélodie et de la perturbation

## déclaration d'une série de hauteurs ordonnée (= pitchset)

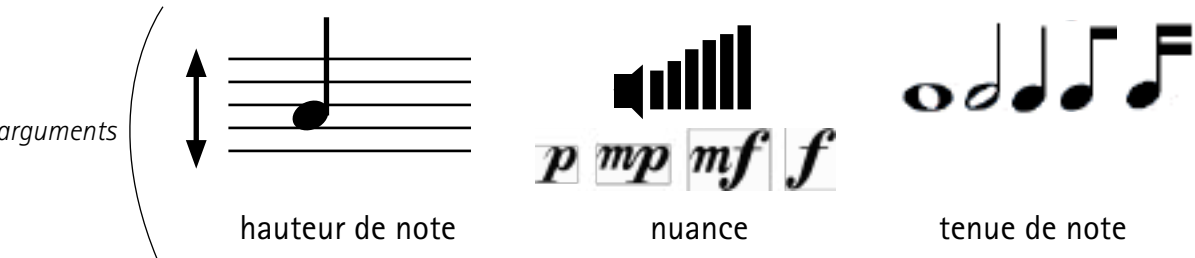
float[] pitchSet =

{57	60	60	60	62	64	67	67	69	72	72	72	74	76	79};
<b>la</b>	<b>do</b>	<b>do</b>	<b>do</b>	<b>ré</b>	<b>mi</b>	<b>Sol</b>	<b>Sol</b>	<b>LA</b>	<b>DO</b>	<b>DO</b>	<b>DO</b>	<b>RE</b>	<b>MI</b>	<b>SOL</b>

MIDI note number : 0=>127, Do octave-2 => Sol octave 8

## fonction .playnote() jouer une note

void playNote(double pitch, double dynamic, double duration)



type variable **double = taille de chiffre 2x plus grand que float**

mélodie **sc**.playNote (pitchSet [(int)mouseX/100] +keyRoot, random(90)+30, random(20)/10 + 0.2)

perturbation **sc3**.playNote (mouseX/10, random(90)+30, random(20)/10 + 0.2)

```
float[] pitchSet = {57, 60, 60, 60, 62, 64, 67, 67, 69, 72, 72, 72, 74, 76, 79};
float setSize = pitchSet.length;
float keyRoot = 0;
```

```
void draw()
{
  if (mousePressed == true) {
    sc3.playNote(mouseX/10, random(90)+30, random(20)/10 + 0.2);
  }

  if ((random(1) < density) && (crasse==true)) {
    sc.playNote(pitchSet[(int)mouseX/100]+keyRoot, random(90)+30, random(20)/10 + 0.2);

    sc2.playChord(pitches, random(50)+30, 4.0);
  }
}
```

# Fonction musicale (suite)

création de la basse

## déclaration d'une liste de hauteurs (= pitches)

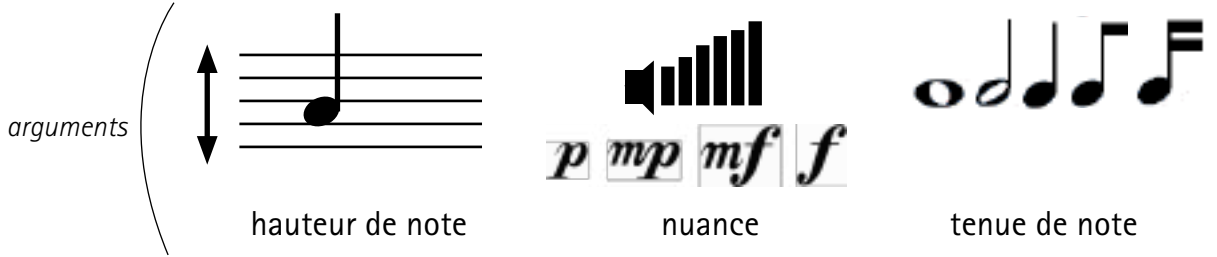
```
float[] pitches =


|                               |                                 |
|-------------------------------|---------------------------------|
| pitchSet[(int) mouseY/100]-12 | pitchSet(int)random(setSize)-12 |
|-------------------------------|---------------------------------|



```

## fonction .playChord() jouer un accord

```
void playChord(float[] pitch, double dynamic, double duration)
```



type variable **double & float**

```
basse .playChord(pitches, random(50)+30, 4.0);
```

```
float[] pitchSet = {57, 60, 60, 60, 62, 64, 67, 67, 69, 72, 72, 72, 74, 76, 79};
float setSize = pitchSet.length;
float keyRoot = 0;
```

```
void draw()
{
  if ((random(1) < density) &&
      (crasse==true)) {
    float[] pitches = {pitchSet[(int)
      mouseY/100]+keyRoot-12, pitchSet[(int)
      random(setSize)]-12};
```

```
sc2.playChord(pitches, random(50)+30, 4.0);
```

# fabrication du visuel

---

## colorMode(HSB) Hue,Saturation,Brightness

remplace le mode RVB (par défaut) par TSL: Teinte, Saturation, Brillance

## le quadrilatère

### rotation

mettre en rotation son centre dans les coordonnées sont (width/2,height/2);

### coloration capteurY

créer une variable incluant le capteur puis l'insérer dans la teinte pour la modifier

### dimensions capteurX

créer une variable incluant le capteur puis l'insérer dans 2 coordonnées définissant les point du quadrilatère

## le cercle

### rotation

mettre en rotation son centre dans les coordonnées sont (width/2,height/2);

### coloration capteurY

créer une variable «r» incluant le capteur puis l'insérer dans la teinte pour la modifier

### dimensions capteurX

créer une variable «taille» incluant le capteur puis l'insérer dans les 2 emplacement définissant la hauteur et la largeur de l'ellipse pour obtenir un cercle

### limiter les dimensions

mettre une condition en cas de dépassement des limites supérieures et inférieures

```
float x;  
float y;  
float a=0; //angle de départ  
float b=0; //angle de départ  
float l=10; //largeur ellipse de départ  
float h=10; //hauteur ellipse départ
```

```
void setup()  
{
```

```
  size(1080, 700);  
  background(250, 250, 250);  
  colorMode(HSB);  
  smooth();  
  noStroke();
```

```
void draw()  
{
```

```
  translate(width/2,height/2);  
  rotate(b);
```

```
  float r=mouseY;  
  fill(r+20,random(210),255);
```

```
  float f= random(mouseX/10);  
    quad(10,35,f,10*f,75,80,30*f,65);  
    b+=0.4;  
  smooth();  
  noStroke();
```

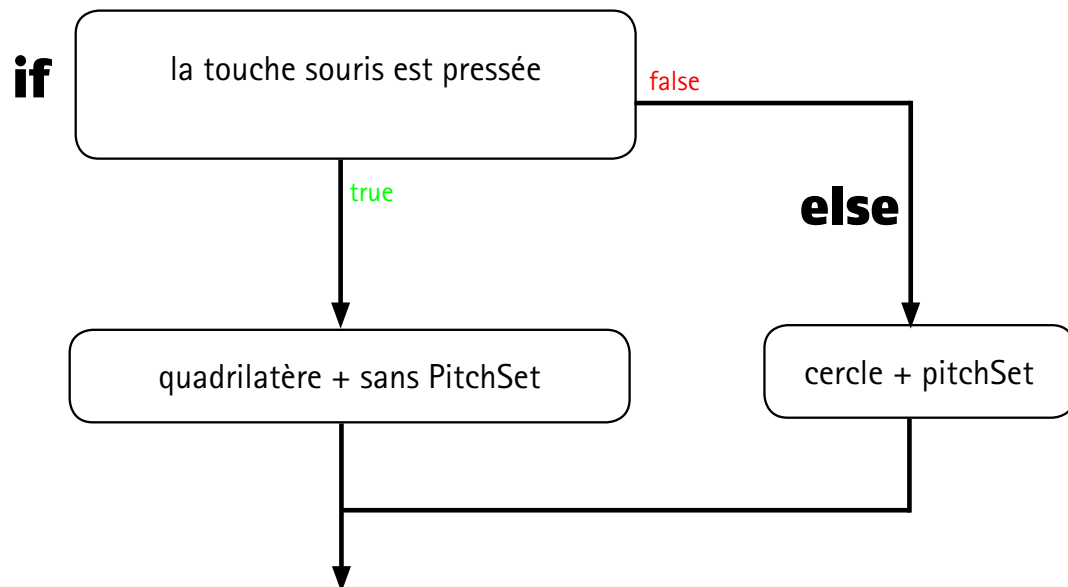
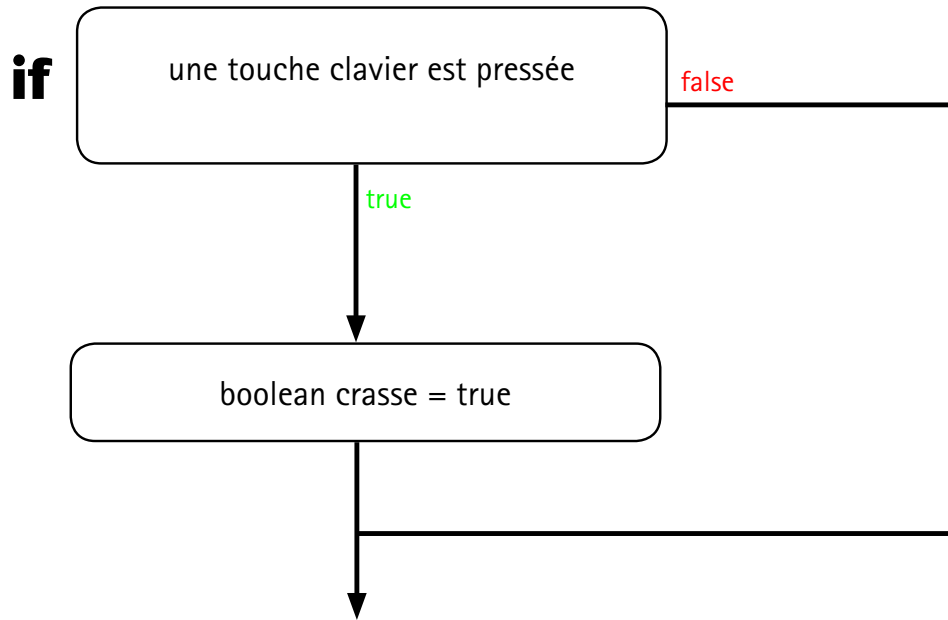
```
  translate(width/2,height/2);  
  rotate(a);
```

```
    float r=mouseY;  
    if (r<30){ //taille supérieure limite  
      r = 50;  
    }
```

```
      fill(r,random(190),random(200));  
  float taille = mouseX ;  
  if (taille > 350){ //taille supérieure limite  
    taille = 350;  
  }  
  if (taille < 50){ //taille inférieure limite  
    taille = 50;  
  }  
  ellipse(300, 0, taille, taille);  
  a+=0.1;
```

## structures conditionnelles

Après l'ouverture de la fenêtre je ne voulais pas que mon programme démarre seul. J'ai du placer des booléens.



```
boolean crasse = false ;
```

```
void draw()  
{
```

```
    if (keyPressed == true){  
        crasse = true;  
    }
```

```
    if (mousePressed == true) {  
        sc3.playNote(mouseX/10, random(90)+30,  
random(20)/10 + 0.2);  
        translate(width/2,height/2);  
        rotate(b);  
        float r=mouseY;  
        fill(r+20,random(210),255);  
        float f= random(mouseX/10);  
        quad(10,35,f,10*f,75,80,30*f,65);  
        b+=0.4;  
    }
```

```
else {
```

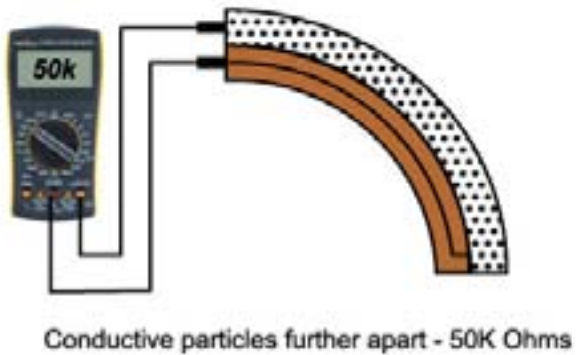
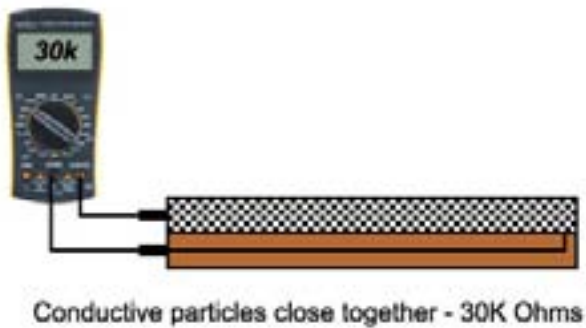
```
    if ((random(1) < density) &&  
        (crasse==true)) {  
        float[] pitches = {pitchSet[(int)  
mouseY/100]+keyRoot-12, pitchSet[(int)  
random(setSize)]-12};
```

```
        sc.playNote(pitchSet[(int)mouseX/100]+keyRoot,  
random(90)+30, random(20)/10 + 0.2);
```

```
        sc2.playChord(pitches, random(50)+30, 4.0);
```

```
    if(crasse == true) {  
        translate(width/2,height/2);  
        rotate(a);  
        //////////[...]//////////  
        ellipse(300, 0, taille, taille);
```

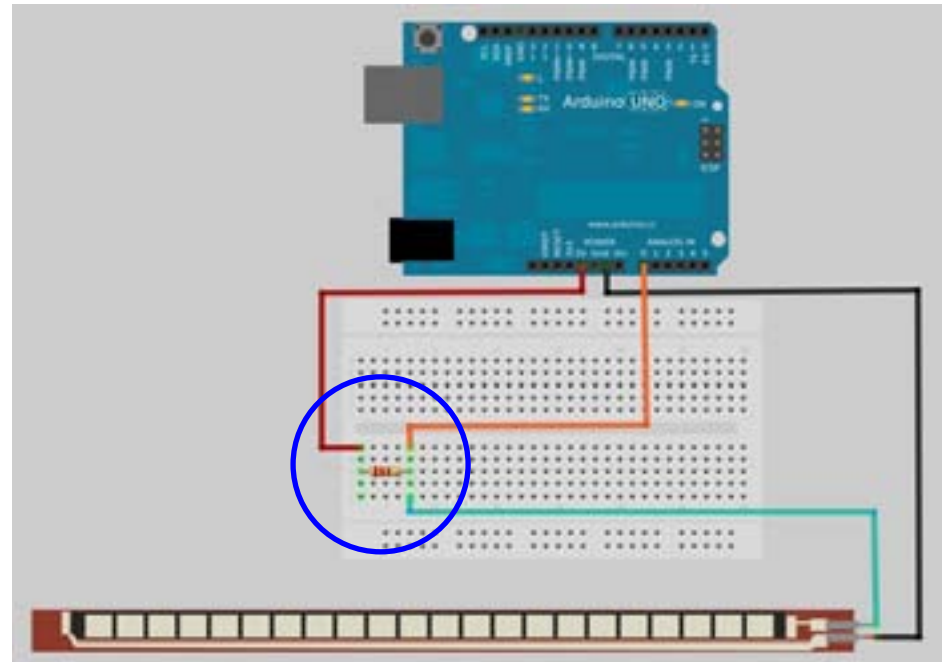
## résistance variable



= potentiomètre ?

## pont diviseur de tension!

divise la tension d'entrée (5V)  
grâce à une résistance fixe  
pour créer une tension de référence



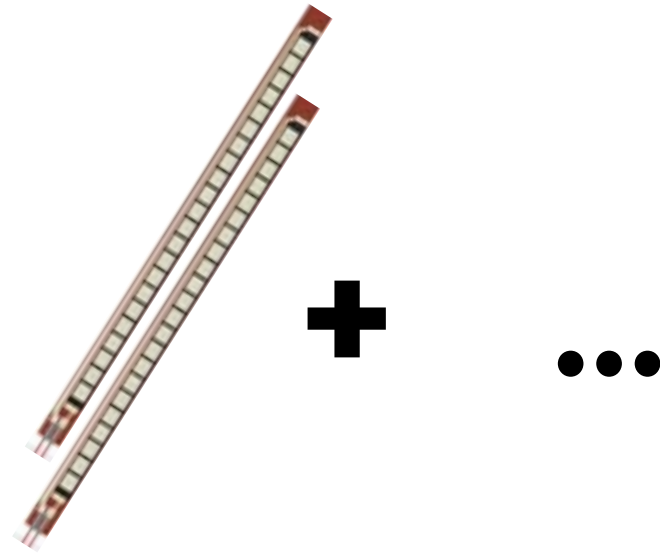
données inutilisables  
dans serialCallResponse

Pour la suite

---



**mieux contrôler SoundCipher  
ou  
changer d'environnement**



**ajouter périphériques graphiques**

# POUR FINIR...

## Typographies

Rotis avec/sans serif, Otl Atcher, 1988

Fago, Ole Schäfer, 2000

Lucida console, Charles Bigelow, 1985