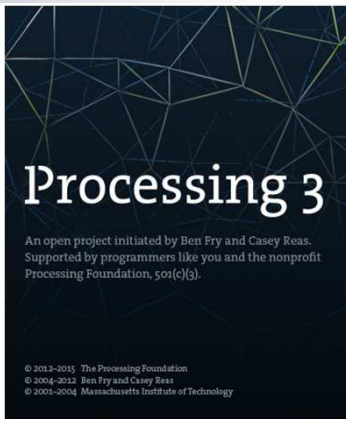
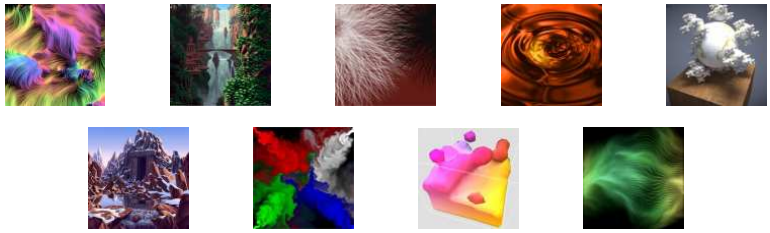


III – Le développement sous « Processing »

30

III.1 – Présentation de Processing

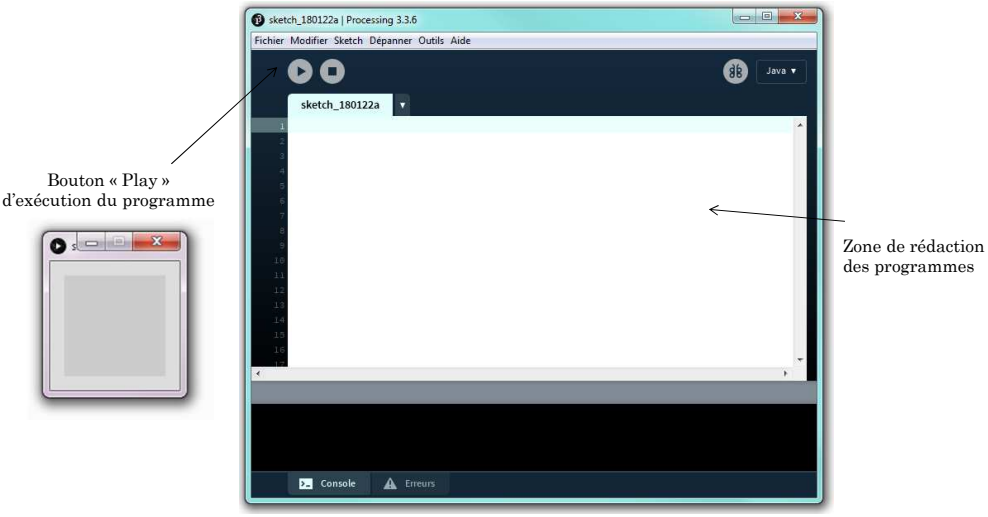
- P55, Proce55ing, ...
  - « Open-source language and environment for learning the fundamentals of electronic art and computer programming »
  - Processing est un logiciel permet de programmer dans un langage très simple (dérivé du JAVA et du C) des expériences de Réalité Virtuelle...
- Outil intégralement gratuit !
  - Site officiel de Processing : https://www.processing.org/
  - Beaucoup d'exemples sur le site officiel, mais également ici : https://www.openprocessing.org/
- Quelques résultats d'exécution de **programmes** d'exemples :



31

III.1 – Interface de l'outil de développement

- Un double clic sur l'icône du programme ouvre la fenêtre suivante :



- Un programme en processing s'appelle un « sketch »

32

III.1 – Premier « sketch »

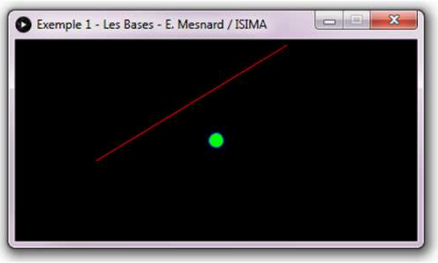
- Exemple de programme, traçant une ligne et un cercle

```
/*****************************************************************************/
/*  MESNARD Emmanuel                                              ISIMA      */
/*                                                                           */
/*          Exemple 1 : ouverture d'une fenetre et trace d'une ligne         */
/*                                                                           */
/* Exemple_1_Les_Bases.pde                                 Processing 3.0    */
/*****************************************************************************/

// Declarations de constantes : Quelques couleurs...
final int rouge = color(255,0,0);
final int vert  = color(0,255,0);
final int bleu  = color(0,0,255);
final int noir  = color(0,0,0);
final int blanc = color(255,255,255);

// Fonction d'initialisation de l'application - executee une seule fois
void setup() {
  // Initialisation des parametres graphiques utilises
  size(400,200); // Fenetre de 400*200, sans appeler la carte graphique
  // size(400,200,P2D); // Fenetre de 400*200 avec acceleration carte graphique, en 2D
  surface.setTitle("Exemple 1 - Les Bases - E. Mesnard / ISIMA");
```

33

III.1 – Premier « sketch », suite et fin

```
  colorMode(RGB,255,255,255); // fixe format couleur R G B pour fill, stroke, etc...
  fill(vert);                 // couleur remplissage RGB - noFill() si pas de remplissage
  stroke(rouge);              // couleur pourtour RGB - noStroke() si pas de pourtour
  background(noir);           // couleur fond fenetre
  line(80,120,270,5);         // trace d'une ligne rouge : x=largeur; y=hauteur (vers le bas!)
  stroke(bleu);               // changement de couleur pour trace un cercle bleu...
  ellipse(width/2,height/2,15,15); // au centre de la fenetre, de rayon 15
}

// Fonction de re-tracage de la fenetre - executee en boucle
void draw() {
  // Ne rien faire de particulier !
}
```



Choix de la largeur du trait :
`strokeWeight()`

Autres tracés possibles :
`triangle()`
`line()`
`arc()`
`point()`
`quad()`
`ellipse()`
`rect()`

34

III.1 – Premières notions à retenir...

- Deux fonctions **essentielles** :
  - une fonction `setup()` exécutée une unique fois, au démarrage
  - une fonction `draw()` exécutée en boucle, à la vitesse maximale du processeur
- des fonctions pour le paramétrages des fenêtres :
  - `size(640,480)` : la taille de la fenêtre de l'application
  - `surface.setTitle()` : le titre
  - `background()` : couleur de fond de la fenêtre
- des fonctions pour les tracés graphiques :
  - `colorMode(RGB,255,255,255)` : choix du format de gestion des couleurs
  - `fill()` : couleur de remplissage, et `noFill()` si pas de remplissage
  - `stroke()` : couleur du trait, et `noStroke()` si pas de pourtour
  - couleurs disponibles
    - sélectionnables directement à l'aide de l'outil « sélecteur de couleur »
    - noms réservés, en constantes standard (voir fichier : « `standard_colors.pde` »)
- des fonctions de dessin (primitives 2D) :
  - `triangle()`, `line()`, `arc()`, `point()`, `quad()`, `ellipse()`, `rect()`

35

III.2 – Gestion du clavier par Processing

- **Interactions « classiques »** : souris et clavier
- Principales informations et fonctions associées au clavier :
  - Informations sur le « code » de la touche
    - `key` : nom de la touche, par exemple 'a' pour la première lettre de l'alphabet
    - `keyCode` : même chose, sous la forme d'un code ASCII (valeur 65 pour la lettre a)
    - Certaines touches « spéciales » ne sont repérables que par leur code ASCII :
      - LEFT : 37, UP : 38 , RIGHT : 39, DOWN : 40
      - SHIFT : 16
      - CONTROL : 17
      - ALT : 18
      - PAGE UP (page haute) : 33
      - PAGE DOWN (page basse) : 34
  - Information d'état d'une touche
    - `keyPressed` : vaut `true` si une des touches est enfoncée
  - Fonctions événementielles
    - `keyPressed()` : fonction invoquée automatiquement à l'instant où une touche est enfoncée
    - `keyReleased()` : invoquée lorsque la touche est relâchée
    - `keyTyped()` : invoquée lors d'un cycle d'appui suivi d'un relâchement
    - Attention au Focus : il faut cliquer sur la fenêtre Windows pour que les événements d'appui des touches soient envoyés aux fonctions événementielles de processing !

36

III.2 – Exemple 2 : image virtuelle par clavier (1/4)

```
/*****************************************************************************/
/*  MESNARD Emmanuel                                              ISIMA      */
/*                                                                           */
/*          Exemple 2 : En route vers la Virtualite                          */
/*                      Recuperation des informations "clavier"              */
/*                                                                           */
/* Exemple_2_Clavier.pde                                  Processing 3.0     */
/*****************************************************************************/

// Declarations de constantes : Quelques couleurs...
final int rouge = color(255,0,0);
final int vert  = color(0,255,0);
final int bleu  = color(0,0,255);
final int noir  = color(0,0,0);
final int blanc = color(255,255,255);

// Declarations des variables
int X,Y;   // Coordonnees des points a tracer
int Delta; // Le "pas" d'incrementation
boolean flagDebug = true; // Booleen pour afficher ou non les infos de debug
```

37

III.2 – Exemple 2 : image virtuelle par clavier (2/4)

```
// Fonction d'initialisation de l'application - executee une seule fois
void setup() {
  // Initialisation des parametres graphiques utilises
  size(400,200);
  surface.setTitle("Exemple 2 - Le Clavier - E. Mesnard / ISIMA");
  colorMode(RGB, 255,255,255); // fixe format couleur R G B pour fill, stroke, etc...
  noFill(); // pas de remplissage
  stroke(rouge); // couleur pourtour RGB - noStroke() si pas de pourtour
  background(blanc); // couleur fond fenetre
  smooth(); // Activation de l'anti-aliasing (smooth)

  // Initialisation des variables
  X=0;
  Y=0;
  Delta=1;
} // Fin de Setup

// Fonction de re-tracage de la fenetre - executee en boucle
void draw() {
  // Rien de particulier !
}
```

38

III.2 – Exemple 2 : image virtuelle par clavier (3/4)

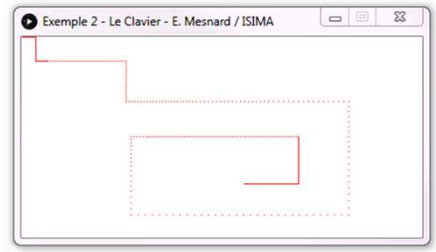
```
// Fonction de gestion des evenements du clavier

void keyPressed() {
  // Analyse des caracteres "classiques"
  switch (key) {
    case 'd' : flagDebug = !flagDebug; // Debug on/off
               break;
    case ' ' : background(blanc); // Appui sur la Barre d'espace
               break;             // pour effacer la fenetre

    // Appui sur les touches plus et moins ...
    case '+' : Delta = (Delta<15) ? (Delta+1) : 15; // ... pour incrementer le pas
               break;
    case '-' : Delta = (Delta>1) ? (Delta-1) : 1; // ... pour decrementer le pas
               break;
  }

  // Analyse des caracteres "etendus"
  switch (keyCode) {
    // Fleches : LEFT, RIGHT, UP, DOWN
    case LEFT    : X-=Delta;
                   X = (X<0) ? 0 : X;
                   break;
    case RIGHT   : X+=Delta;
                   X = (X>width) ? width : X;
                   break;
```

39

III.2 – Exemple 2 : image virtuelle par clavier (4/4)

```
    case UP      : Y-=Delta;
                   Y = (Y<0) ? 0 : Y;
                   break;
    case DOWN    : Y+=Delta;
                   Y = (Y>height) ? height : Y;
                   break;
    // Autres Touches speciales : CONTROL, ALT et SHIFT
  }
  if (flagDebug) {
    println("Touche " + key + " avec le code " + keyCode);
  }
  // Trace d'un point a l'endroit indique
  point(X,Y);
}
```

```
Touche a avec le code 65
Touche b avec le code 66
Touche c avec le code 67
```



40

III.3 – Gestion d'une souris par Processing

- Principales informations et fonctions de la souris :
  - Informations de position de la souris
    - `mouseX` et `mouseY` : position courante
    - `pmouseX` et `pmouseY` : position précédente
  - Information d'état des boutons de la souris
    - `mousePressed` : vaut `true` si un des boutons est enfoncé
    - `mouseButton` : nom du bouton enfoncé : LEFT, CENTER, RIGHT
  - Fonctions événementielles
    - `mousePressed()` : fonction invoquée à l'instant où un bouton de la souris est appuyé
    - `mouseReleased()` : même chose, lorsque le bouton de la souris est relâché
    - `mouseClicked()` : invoquée après un cycle d'appui suivi d'un relâchement
    - `mouseMoved()` : invoquée chaque fois que la souris a bougé
    - `mouseDragged()` : même chose avec clic maintenu
    - `mouseWheel()` : invoquée lors d'une rotation de la molette centrale

41

III.3 – Exemple 3 : image virtuelle par souris (1/3)

```
/*****************************************************************************/
/*  MESNARD Emmanuel                                              ISIMA      */
/*                                                                           */
/*          Exemple 3 : En route vers la Virtualite                          */
/*                      Creation d'une image virtuelle via la souris         */
/*                                                                           */
/* Exemple_3_Souris.pde                                 Processing 3.0       */
/*****************************************************************************/

// Declarations de constantes : Quelques couleurs...
final int rouge = color(255,0,0);
final int vert  = color(0,255,0);
final int bleu  = color(0,0,255);
final int noir  = color(0,0,0);
final int blanc = color(255,255,255);

// Declaration d'une variable globale
int totalMolette; // Gestion de la roulette centrale

// Fonction d'initialisation de l'application - executee une seule fois
void setup() {
  // Initialisation des parametres graphiques utilises
  size(640,480,P2D);  // Avec acceleration graphique 2D
  surface.setTitle("Exemple 3 - La souris - E. Mesnard / ISIMA");
```

42

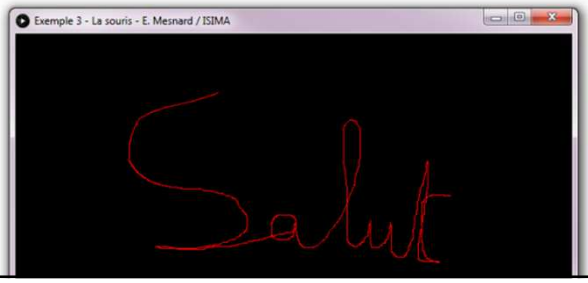
III.3 – Exemple 3 : image virtuelle par souris (2/3)

```
  colorMode(RGB, 255,255,255); // fixe format couleur R G B pour fill, stroke, etc...
  noFill(); // pas de remplissage
  stroke(rouge); // couleur pourtour RGB - noStroke() si pas de pourtour
  background(blanc); // couleur fond fenetre
  smooth(); // Activation de l'anti-aliasing (smooth)

totalMolette = 0;
} // Fin de Setup

// Fonction de re-tracage de la fenetre - executee en boucle
void draw() {
  if (mousePressed && (mouseButton == RIGHT)) { // Test clic droit de la souris...
    stroke(vert);    // ... pour trace un rond vert en plus des lignes
    ellipse(mouseX, mouseY, 5, 5);
  }
}

// Fonctions de gestion des evenements de la souris
void mousePressed() { // Fonction invoquee lors de l'appui sur un des deux boutons
  background(noir);   // Equivaut a effacer la fenetre et mettre tout en noir
}

void mouseReleased() { // Fonction invoquee lors du relachement d'un bouton
  background(blanc);   // Equivaut a effacer la fenetre et mettre tout en blanc
}
```

43

III.3 – Exemple 3 : image virtuelle par souris (3/3)

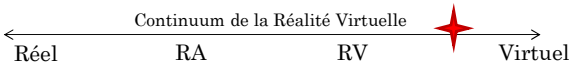
```
void mouseDragged() { // Fonction invoquee tant que le bouton est maintenu appuye
  stroke(rouge);      // Trace d'une ligne rouge entre la nouvelle position de
  line(pmouseX, pmouseY, mouseX, mouseY); // la souris et l'ancienne
}

void mouseWheel(MouseEvent event) {
  totalMolette += event.getCount(); // Prise en compte de l'evenement
  println(totalMolette); // Affichage retour console pour debug
  background(blanc); // Effacement de la fenetre
  // Affichage des informations sur la fenetre de l'application
  text("Total du decalage molette centrale : ",10,20);
  text(str(totalMolette),222,20);
}
```



44

TD 1 Réalité Virtuelle : Fluide psychédélique

- Du REEL dans beaucoup de VIRTUEL



  - Projet « MouseFluid » permettant le brassage de fluides colorés

  - Les mouvements réels sur la souris impactent le monde virtuel 2D
    - A noter : Utilisation d'un solveur de mécanique des fluides 2D (« MSA Fluid »)
- Travail demandé :
  - Analyser le code fourni...
  - ... intégrer la **molette** : la valeur de la molette doit modifier le *facteurCouleur*
  - ... et intégrer le **clic droit** : flag de gestion du tracé couleur

45