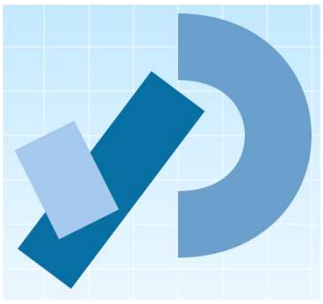


Processing 4

An open project initiated by Ben Fry and Casey Reas.
Supported by the community and the nonprofit
Processing Foundation 501(c)(3).

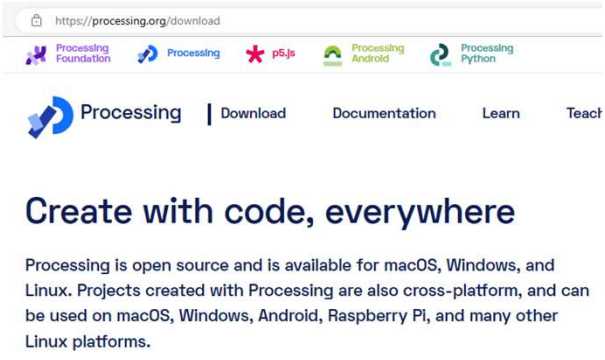
III – Le développement sous « Processing »

27

27

III.1 – Présentation de Processing

- Processing (P55, Proce55ing, ...)
  - Outil logiciel dédié au dessin artistique, statique ou dynamique…
  - description par suites d'instructions : programme écrit dans un langage de « scripting » simplifié (dérivé du JAVA)
- Outil gratuit et multiplateforme
  - Site officiel de Processing : https://www.processing.org/

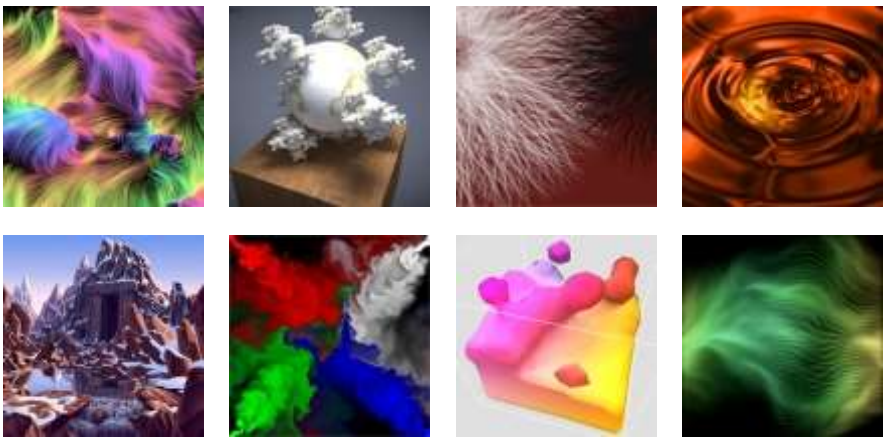


28

28

III.1 – Présentation de Processing

- Quelques résultats d'exécution de **programmes** d'exemples :



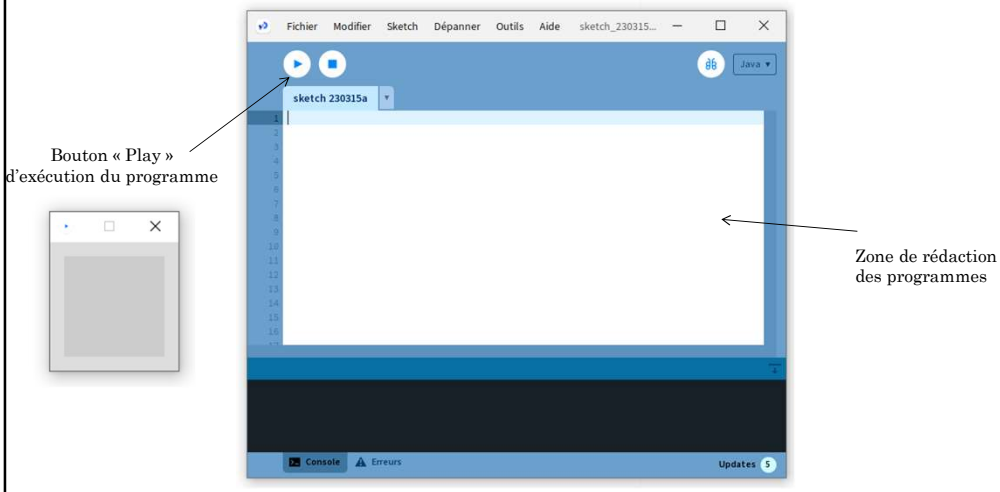
- Un programme en processing s'appelle un « sketch » (croquis)

29

29

III.1 – Interface de l'outil de développement

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



Bouton « Play » d'exécution du programme

Zone de rédaction des programmes

En standard, un projet Processing = fichier (d'extension PDE) dans un répertoire du **même nom !**

30

30

III.1 – Ecriture d'un programme – les bases

- 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 (dans `setup` uniquement, et avec constantes). A noter : `width`, `height`
  - `surface.setTitle()` : le titre (dans `setup` uniquement)
  - `background()` : couleur de fond de la fenêtre

31

31

III.1 – Premier « sketch » : Exemple 1

- Programme ouvrant une fenêtre avec fond noir

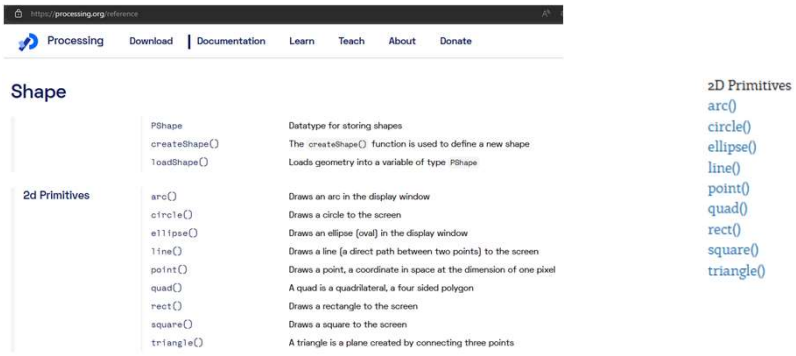
```
/*****************************************************************************/
/*  MESNARD Emmanuel                                              ISIMA      */
/*                                                                           */
/*              Les bases : Ouverture d'une fenetre                          */
/*                                                                           */
/* Exemple_1_Fenetre_Titre.pde                          Processing 4.2       */
/*****************************************************************************/

// Fonction d'initialisation de l'application - executee une seule fois
void setup() {
  // Initialisation des parametres graphiques utilises
  size(600,300); // Fenetre de 600*300, sans appeler la carte graphique
  // size(600,300,P2D); // Fenetre de 600*300 avec acceleration carte graphique, en 2D
  surface.setTitle("Exemple 1 Basique : Fenetre avec titre - E. Mesnard / ISIMA");
  background(0); // Fond noir
}

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

32

32

III.1 – Les tracés graphiques : formes et couleurs

- des fonctions de dessin (primitives 2D de tracé) :
  - `line()`, `point()`, `circle()`, `ellipse()`, `square()`, `rect()`, ...
  - Voir l'aide (en ligne : https://processing.org/reference ou clic droit sur un nom de fonction)



- avec ou sans pourtour et/ou remplissage :
  - `stroke()` : avec tracé du trait, et `noStroke()` pas de pourtour
  - `strokeWeight()` : épaisseur du trait
  - `fill()` : remplissage coloré, et `noFill()` pas de remplissage

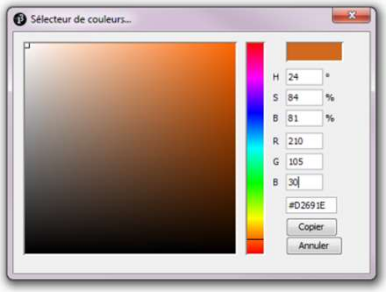
33

33

III.1 – La gestion des couleurs

- Format d'un pixel, sous Processing :
  - Type Couleur : nombre entier `int` ou `color`
  - 3 x 8 bits, donc (255x255x255 couleurs possibles)

  - ou encore : 4 x 8 bits, avec une composante Alpha (transparence)
  - Deux modes de gestion des couleurs :
    - RGB : « Red », « Green », « Blue »
    - HSB : « Hue », « Saturation », « Brightness »
  - `colorMode(RGB,255,255,255)` : choix du format de gestion des couleurs
- Sélecteur de couleurs (menu « outils ») :



- Exemple en mode RGB :

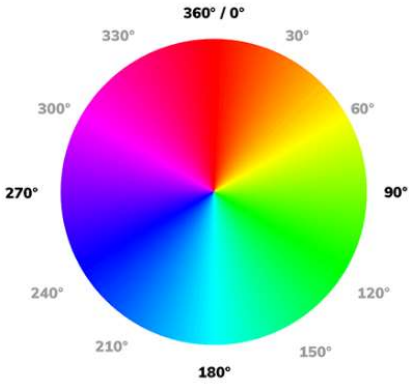
```
colorMode(RGB,255,255,255);
color chocolate = color(210,105,30);
// equivalent a :     chocolate = color(0xD2,0x69,0x1E);
//                    chocolate = #D2691E;
//                    chocolate = 0xFFD2691E; // 0xFF composante Alpha
```

34

34

III.1 – La gestion des couleurs

- Roue « chromatique » de l'HSB :
  - « Hue » (H, Teinte) : de 0 à 360°
  - « Saturation » (S) : 0 à 100%
  - « Brightness » (B, luminosité) : 0 à 100%



- Exemple en mode HSB :

```
color chocolate = color(210,105,30); // en RGB
colorMode(HSB, 360, 100, 100);       // changement de mode
float teinte = hue(chocolate);       // ici, teinte = 25.00
```

35

35

III.1 – Exemple 2 : Formes géométriques colorées

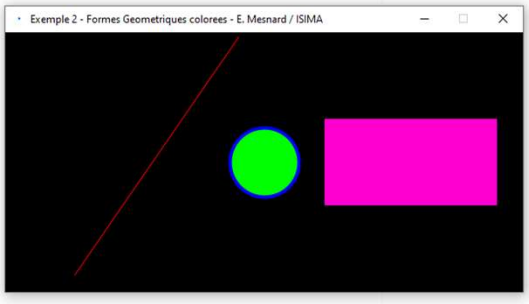
```
/*****************************************************************************/
/* Exemple_2_Formes_Geometriques.pde                    Processing 4.2   */
/*****************************************************************************/

// 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);

int PseudoTeinte; // Variable utile pour la gestion du dynamisme

void setup() {
  size(600,300); // Fenetre de 600*300, sans appeler la carte graphique
  surface.setTitle("Exemple 2 - Formes Geometriques colorees - E. Mesnard / ISIMA");
  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(80,280,270,5);          // trace d'une ligne rouge
  stroke(bleu);                // changement de couleur pour tracer un cercle bleu...
  strokeWeight(4);             // avec pourtour epais
  circle(width/2,height/2,80); // au centre de la fenetre, de rayon 80
```

36

36

III.1 – Exemple 2, suite et fin

```
  colorMode(HSB,360,100,100); // changement de mode
  noStroke();                 // sans pourtour
  PseudoTeinte = 0;           // initialization de la variable globale en fin de setup
}

// Fonction de re-tracage de la fenetre - executee en boucle
void draw() {
  // Trace d'un rectangle uni, mais de couleur (teinte) changeante
  fill(color(PseudoTeinte%360,100,100));
  rect(370,100,200,100);
  PseudoTeinte++;
}
```



37

37

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 :la fenêtre doit être au **premier plan** (« Windows focus ») pour que les événements d'appui des touches soient envoyés aux fonctions événementielles de Processing !

38

38