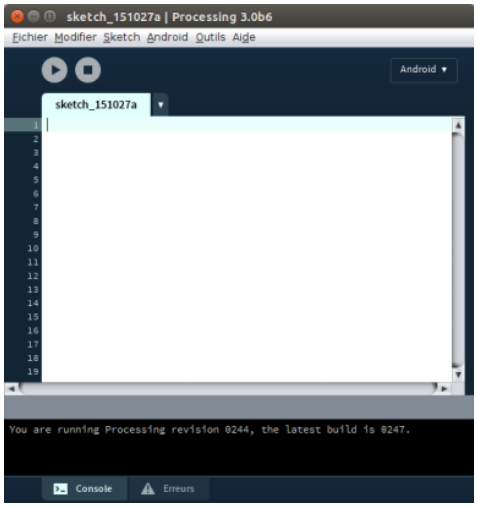
1. QU'EST-CE QUE PROCESSING ?



Processing est un **langage** et un **environnement de programmation** open-source. Il donne la possibilité de créer des **interfaces graphiques** interactives et notamment des interfaces permettant la communication avec le système Arduino. On peut ainsi commander à la souris ou au clavier la carte Arduino ou bien afficher sur le PC des graphiques à partir des données reçues depuis la carte Arduino. Processing permet également de réaliser de la capture vidéo, du traitement d'image, des applications réseau serveur ou client, de la lecture de son, de la reconnaissance vocale... !!

Processing est un **logiciel libre** (open-source) totalement gratuit pour l'utilisateur. Le logiciel fonctionne sur Macintosh, sous Windows et sous Linux, en effet il est basé sur la plate-forme Java. Les programmes réalisés avec Processing peuvent être lus par les navigateurs internet équipés du plug-in java, mais aussi sous forme d'applications indépendantes pour Windows, Linux ou Mac (en réalité n'importe quelle machine disposant d'une Machine virtuelle Java).

L'interface de Processing est très proche de celle du logiciel Arduino.

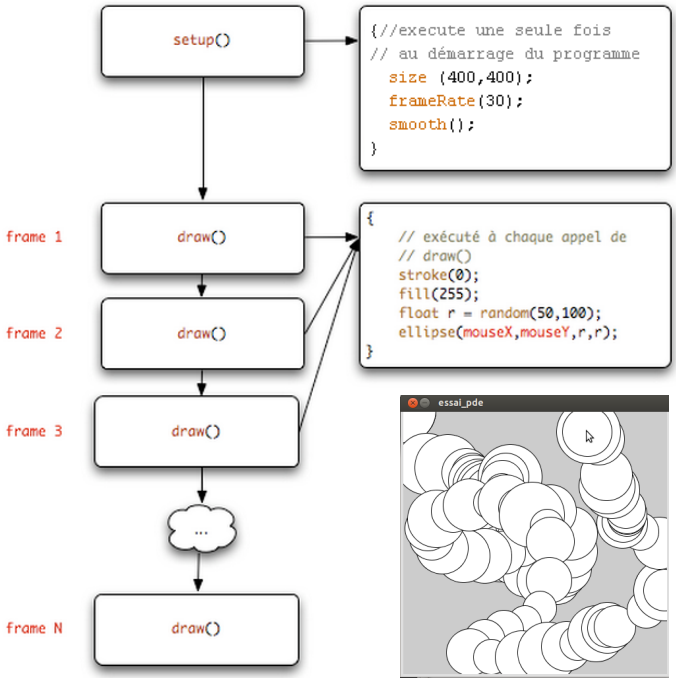
2. LE LANGAGE DE PROGRAMMATION

Le langage de base de processing est le **Java**, auquel est ajouté des fonctions propres à Processing. Un programme contient un code minimal de **2 fonctions** :

```
void setup()   //fonction d'initialisation de la carte
{

}

void draw()    //fonction principale, elle se répète à l'infini

{

}
```

La fonction **setup()** est appelée au démarrage du programme. Cette fonction est utilisée pour initialiser les variables, pour préparer la fenêtre de visualisation (l'espace de dessin), par exemple en lui donnant une taille au départ. La fonction setup() n'est exécutée qu'une seule fois au lancement du programme.

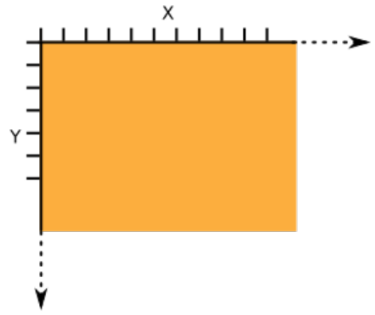
La fonction **draw()** s'exécute en boucle infinie, permettant de créer l'interface graphique. L'exécution de cette boucle s'effectue par défaut au rythme de **30 fois par seconde,** jusqu'à ce que l'utilisateur arrête le programme. La fonction ***frameRate(x)*** permet toutefois de modifier cette valeur (x étant le nombre de répétitions par seconde).



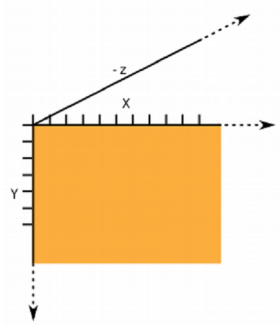
L'espace de dessin

L'espace de dessin constitue la fenêtre de visualisation où s'affichera vos réalisations dans Processing en 2 ou 3 dimensions.

Instruction	Définition
size (largeur,hauteur)	taille de la fenêtre de dessin, par défaut de 100x100 pixels



En 2 dimensions (2D), on utilise deux axes de coordonnées x et y correspondant respectivement à la largeur (axe horizontal) et à la hauteur (axe vertical). Le coin en haut à gauche correspond aux valeurs x=0 et y=0.



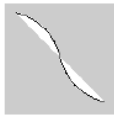
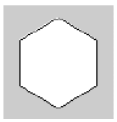
En 3 dimensions (3D), en plus des deux axes de coordonnées, on a un troisième axe de coordonnées z, exprimant la profondeur : dans ce cas précis, on utilise la commande size avec un troisième paramètre indiquant que l'on travaille dans un espace en 3D : size(100, 100, P3D);

Au sein d'un programme, on peut connaître à tout moment la taille de l'espace de dessin utilisé au moyen des mots-clés **width** et **height**. Ces instructions sont très utiles lorsque l'on souhaite notamment dessiner des formes qui puissent s'adapter ultérieurement aux éventuels changements de dimension de la fenêtre de visualisation.

Les formes

Beaucoup de formes prédéfinies sont fournies par Processing. En voici les principales :

Forme	Instruction
le point	point(x, y)
la ligne	line(xA, yA, xB, yB)
le rectangle	rect(x, y, largeur, hauteur)
l'ellipse	ellipse(x, y, largeur, hauteur)
le triangle	triangle(x1, y1, x2, y2, x3, y3)
l'arc	arc(x, y, largeur, hauteur, début, fin)
le quadrilatère	quad(x1, y1, x2, y2, x3, y3, x4, y4) les quatre paires de coordonnées x et y sont données dans le sens horaire.

Forme	Instruction
courbe	curve(x1, y1, x2, y2, x3, y3, x4, y4) x1 et y1 définissent le premier point de contrôle, x4 et y4 le second point de contrôle , x2 et y2 le point de départ de la courbe et x3, y3 le point d'arrivée de la courbe. 
courbe bézier	bezier(x1,y1,x2,y2,x3,y3,x4,y4) 
courbe lissée	beginShape() curveVertex(x1,y1) curveVertex(x2,y2) curveVertex(x3,y3 endShape() on définit plusieurs paires de coordonnées. 
formes libres	beginShape() curveVertex(x1,y1) curveVertex(x2,y2) curveVertex(x3,y3 endShape(CLOSE) on définit plusieurs paires de coordonnées. La fonction CLOSE indique que la figure sera fermée. 
3D : la sphère	sphere(taille) 
3D : la boîte	box(longueur, largeur, profondeur) 

Les couleurs

Instruction	Définition
background(x)	couleur de la fenêtre en niveaux de gris; x de 0 (noir) à 255 (blanc)
background(R,G,B)	couleur de la fenêtre en RGB (de 0 à 255 par couleur)
stroke(x)	couleur de tracé d'une forme en niveaux de gris; x de 0 (noir) à 255 (blanc)
stroke(R,G,B)	couleur de tracé d'une forme en RGB (de 0 à 255 par couleur)
stroke(R,G,B,x)	couleur de tracé d'une forme en RGB (de 0 à 255 par couleur), x opacité de la couleur (de 0 le plus clair à 255 le plus foncé)
strokeWeight(x)	fixe la largeur du trait de tracé en pixels
noStroke()	contours invisibles
fill(x)	couleur de remplissage d'une forme en niveaux de gris; x de 0 (noir) à 255 (blanc)
fill(R,G,B)	couleur de remplissage d'une forme en RGB (de 0 à 255 par couleur)
fill(R,G,B,x)	couleur de remplissage d'une forme en RGB (de 0 à 255 par couleur), x opacité de la couleur (de 0 le plus clair à 255 le plus foncé)
noFill()	couleur de remplissage invisible

Remarque 1 : il est toujours possible de définir les couleurs en notation hexadécimale, par exemple #ffcc33.
Remarque 2 : par défaut, toute modification de style (couleur de remplissage ou de contour, épaisseur ou forme de trait) s'appliquera à tout ce que vous dessinerez ensuite. Pour limiter la portée de ces modifications, vous pouvez les encadrer par les commandes pushStyle() et popStyle().

Le texte

Il est possible d'écrire du texte sur le dessin avec l'instruction « text » :

Instruction	Définition
text (message, x, y)	affiche le message entre " " aux coordonnées x,y

Remarque : la seule possibilité d'effacer ou de modifier un texte est de réécrire par-dessus.

liens utiles:
http://fr.flossmanuals.net/processing/
http://www.mon-club-elec.fr/pmwiki_mon_club_elec/pmwiki.php?n=MAIN.OUTILSProcessing