Processing

Atelier 1 – des carrés et les couleurs

Ouvrir processing

Taper ce premier code :

```
void setup() {
 size(400,400); // taille de la fenetre
 background(0); //couleur de fond de la fenetre
}
```

Exécuter le programme en appuyant sur la touche ▸

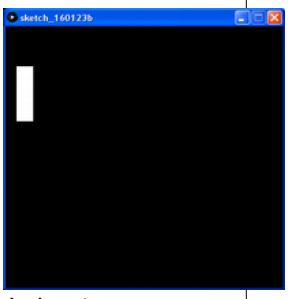
Changer les valeurs de size et de backgroung (0 à 255) et **relancer** le script

L'étape 2 va chercher a faire apparaître un rectangle blanc dans la fenêtre :

```
void setup() {
 size(400,400); // taille de la fenetre
 background(0); //couleur de fond de la fenetre
}

void draw() {
 dessiner(); // appel de la fonction dessiner
}

void dessiner() {
  fill(255);   // fixe la couleur du rectangle suivant
  rect(15,60,25,85); // dessine le rectangle point de départ 15,60 et 25 d'épaisseur et 85 de haut
}
```



Atelier 2 – Interagir avec les formes

Nous allons maintenant chercher à faire bouger le rectangle blanc.
Pour cela il faut rendre la position du point de départ du rectangle variable
On déclare donc 2 variables en tout début de programme (au-dessus du void setup). Ces 2 variables sont globales puisque déclarées pour tout le script.

```
int w, z;
```

Ces 2 variables sont initialisées dans la boucle void setup (entre les accolades) :

```
w = 15 ;
z = 60 ;
```

La ligne rect est également modifiée comme suit :

```
 rect(w,z,25,85); // dessine le rectangle point de départ w,z et 25 d'épaisseur et 85 de haut
```

Lancer le script.
Le résultat est identique.

Nous cherchons maintenant à faire bouger le carré avec la souris. Pour cela, il faut rajouter une fonction « bouger » qui modifie la valeur z en fonction de la position de la souris.

```
void bouger() {
 z = (mouseY);
}
```

Rajouter l'appel de la fonction bouger dans la fonction draw

```
void draw() {
 bouger() ; //appel de la fonction bouger
 dessiner(); // appel de la fonction dessiner
}
```

Lancer le script

Malheureusement lorsque l'on bouge la souris, la trace du carré ne s'efface pas.
Pour regler le problème, rajouter la fonction « nettoyer » et modifier une nouvelle fois la fonction draw pour l'appeler.

```
void draw() {
 nettoyer() ; //Appel de la fonction nettoyer pour enlever les traces
 bouger() ; //appel de la fonction bouger
 dessiner(); // appel de la fonction dessiner
}

void nettoyer() {
background(0);
}
```

Votre code ressemble maintenant à cela :

```
int w, z;

void setup() {
 size(400,400); // taille de la fenetre
 background(0); //couleur de fond de la fenêtre
 w = 15;
 z = 60;
}

void draw() {
 nettoyer() ; //Appel de la fonction nettoyer pour enlever les traces
 bouger() ; //appel de la fonction bouger
 dessiner(); // appel de la fonction dessiner
}

void nettoyer() {
background(0);
}

void dessiner() {
  fill(255);
  rect(w,z,25,85);
}

void bouger(){
  z=(mouseY);
}
```

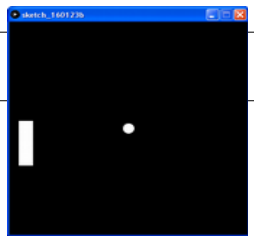
Nous allons maintenant créer un second objet (une balle) qui devra également bouger.
Pour cela il faut déclarer, comme pour le rectangle les variables de position en début de programme et décrire la forme et sa couleur dans la fonction « dessiner » :

```
int x, y;
int w, z;
```

```
void dessiner() {
  fill(255);
  rect(w,z,25,85);
  fill(255);
  ellipse(x,y,20,20);
}
```

et initialiser les variables dans la boucle setup :

```
x = 200;
y = 200;
```



Nous avons maintenant une raquette qui bouge et une balle fixe.

Atelier 3 – Interaction des formes et mouvements automatiques

Pour que la balle bouge nous allons déclarer de nouvelles variables globales et une fonction de déplacement qui intègre le test de rebond sur la raquette.

```
int deplacementX, deplacementY;
int x, y;
int w, z;

void setup() {
 size(400,400); // taille de la fenetre
 background(0); //couleur de fond de la fenêtre
 x = 200;
 y = 200;
 w = 15;
 z = 60;
 deplacementX = 6;
 deplacementY = -3;
}

void draw() {
 nettoyer() ; //Appel de la fonction nettoyer pour enlever les traces
 bouger() ; //appel de la fonction bouger
 dessiner(); // appel de la fonction dessiner
 rebondir() ;// appel de la fonction rebondir
}

void nettoyer() {
background(0);
}

void dessiner() {
  fill(255);
  rect(w,z,25,85);
  fill(255);
  ellipse(x,y,20,20);
  line(200,0,200,400);
}

void bouger() {
 x = x + deplacementX;
 y = y + deplacementY;
 z = (mouseY);
}

void rebondir() {
 if (x > width-10 && deplacementX > 0)  {  // si trop à droite et que le déplacement horizontal est positif
   deplacementX = -deplacementX; // inverser la valeur
 }

 if (y > width-10 && deplacementY > 0)  { // si trop bas et le déplacement vertical est positif
   deplacementY = -deplacementY; // rendre négative la valeur
 }
```

```
if (y < 10 && deplacementY < 10)   {// si on est trop haut et le déplacement vertical est negatif
  deplacementY = abs(deplacementY); // rendre positive cette valeur
}

if (x<w+35 && y>z && y<z+85)  {  // test de la touche de la raquette
  deplacementX = -deplacementX; // inverser la valeur
}

if (x < 10)  { // la balle est derrière la raquette → perdu
 noLoop();
 println("GAME OVER");
}
}
```

Atelier 4 – Difficulté croissante

A - **Rajouter** un compteur qui s'incrémente a chaque fois que la balle rebondie sur la raquette.
Et **afficher** le résultat lorsque le jeu est terminé

B – **Ajouter** une variable de hauteur de la raquette qui diminue a chaque fois que la raquette touche la balle.

C – rajouter un second utilisateur (même rectangle mais avec des touches pour bouger)

```
void keyPressed()
{
   if (key == 'UP')
  {
    z2=z2-10 ;
  }
   if (key == 'DOWN')
  {
    z2=z2+10 ;
  }
}
```