E-ARTSUP.
DÉPARTEMENT DESIGN INTERACTIF & COMMUNICATION VISUELLE.

GRILLES ET RÉPÉTITIONS DE MOTIFS
(STRUCTURES ITÉRATIVES & CONDITIONNELLES)

Digital Lab S.03 // Alexandre Rivaux arivaux@gmail.com

Département Design interactif & communication visuelle:

Enseignants:
Nicolas Baumgartner
Félicie d'Estienne D'Orves
Rémi Jamin
Wolf Ka
Jonathan Munn
Gustave Bernier
Alexandre Rivaux

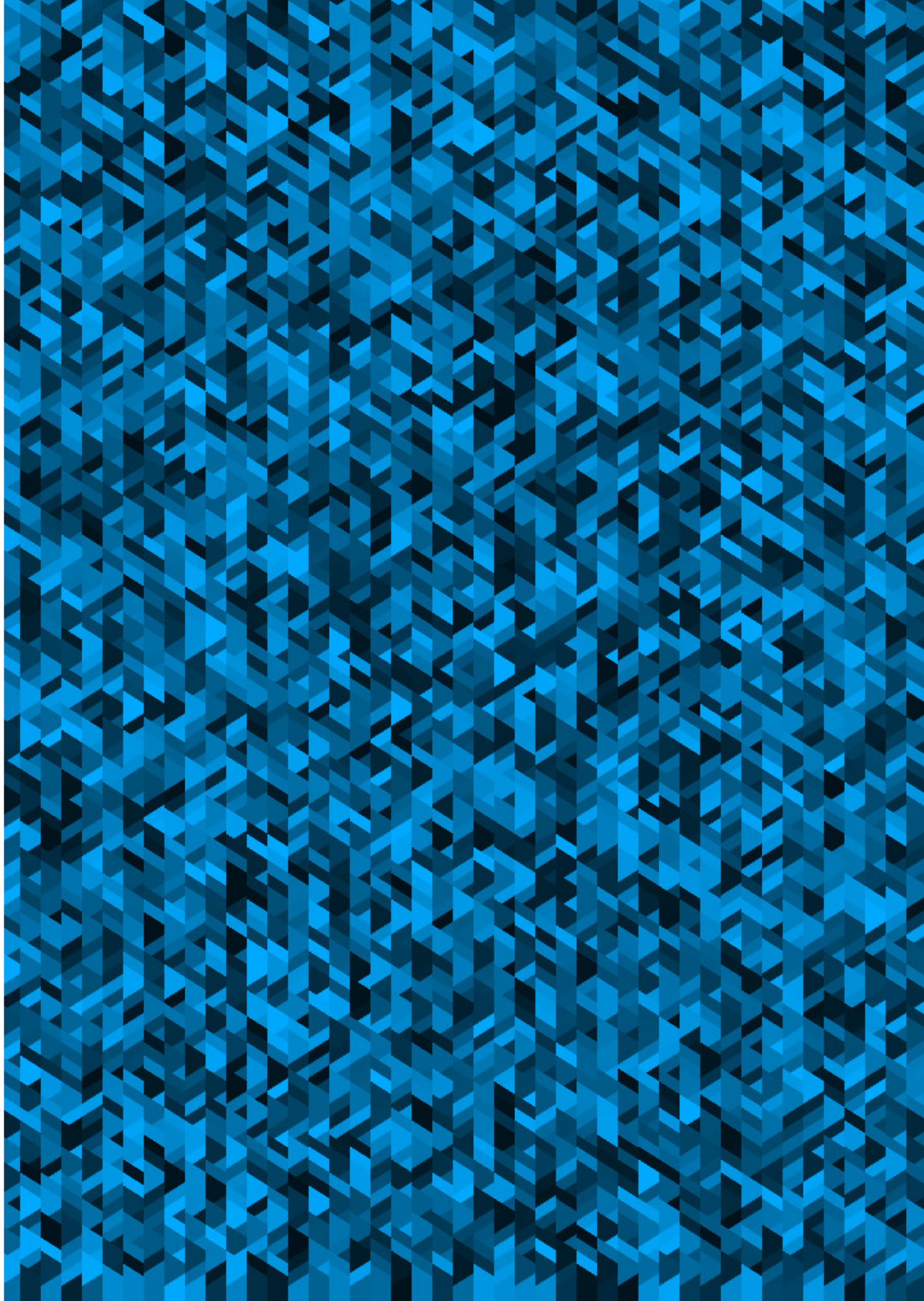
1 - Définitions des structures iteratives & conditionnelles

On appelle ***structure iterative*** une structure qui permet de répéter un certain nombre de fois une série d'instructions simples ou composées. Celle-ci permet notament d'effectuer une grand nombre d'actions ou de calculs simultanés

.

On appelle ***structure conditonnelle*** une structure permettent de tester si une condition est vraie ou non. Cette structure est souvent utilisée afin d'attibuer une valeur à une variable booléenne.

Les ***structures itératives et conditionnelles*** sont deux grande bases de la programmation. Ici nous verrons comment les utiliser afin de créer une série de motif génératif.



2 - Structure itérative : construction

La structure itérative permet d'executer plusieurs fois une ou des instructions. Elle peut s'écrire de plusieurs manières différentes. Ici nous nous concenterons sur la plus connue : ***La boucle for()***. Celle-ci s'écrit de la manière suivante :

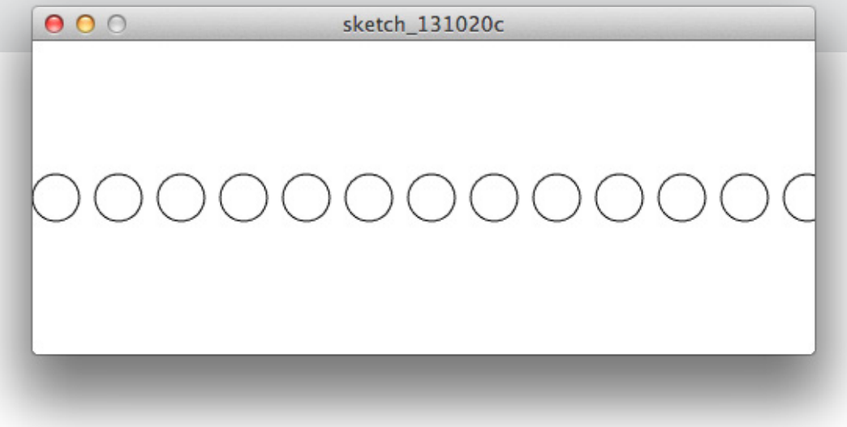
```
for(int i=0; i<Maximum; i++)
{
 instruction
}
```

Où i est le compteur; i est inferieur au nombre maximum de répétitions et i s'incrémente de 1. On peut alors traduire cette phrase de la manière suivante :

Pour i = 0; i étant toujours inférieur à une valeur maximum
et i s'incrémentant de 1 à chaque boucle alors...

Appliquons cela à un exemple concret.
Ici je souhaite déssiner un motif composé de cercles de diamètre 30 pixels et se répétant tout les 40 pixels.

```
for(int i=0; i<width; i+=40)
 {
  ellipse(i+15, height/2, 30, 30);
 }
```



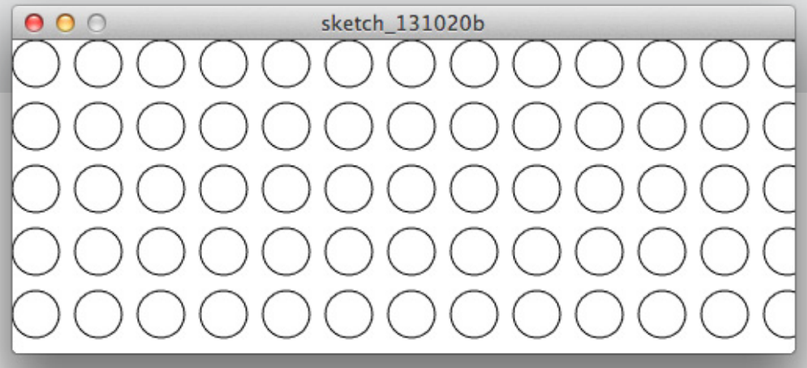
On peut alors traduire cette phrase de la manière suivante :

Pour i=0; i étant toujours inférieur à la largeur de mon sketch; et i s'écrémentant de 40 pixels (0, 40, 80...) je dessine des cercles dont x = i, y = moitié de la hauteur de mon skecth et de diametre 30 pixels.

2 - Structure itérative : construction

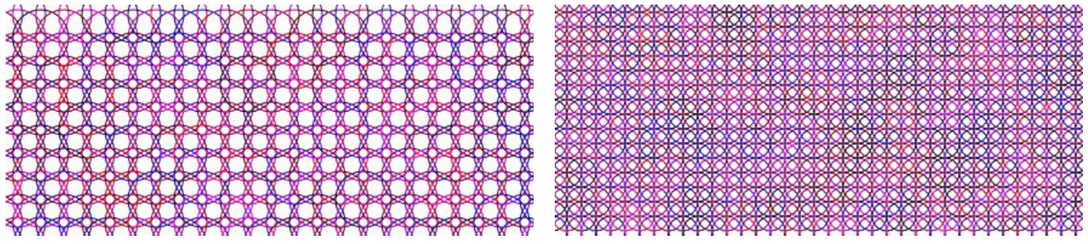
Dans notre premier exemple nous avons utilisé une structure iterative pour répéter notre visuel sur l'axe de x. Dans le cadre d'un motif nous aurons besoins de la répéter à la fois sur les x et les y. Pour cela nous allons utiliser une double boucle for.

```
for (int i=0; i<width; i+=40)
 {
  for (int j=0; j<height; j+=40)
  {
   ellipse(i+15, j+15, 30, 30);
  }
 }
```



On peut alors traduire cette phrase de la manière suivante :

Pour i=0; i étant toujours inférieur à la largeur de mon sketch; et i s'écrémentant de 40 pixels (0, 40, 80...) et pour j = 0; j toujours inférieur à la hauteur de mon sketch et j s'incrémentant de 40 pixels, je dessine des cercles de position i et j et de diametre 30.



3 - Structure conditionnelle : construction

La structure conditionnelle est la structure la plus basique en programmation. Elle permet d'excuter une instruction si un condition est validée. Il existe plusieurs structures conditionnelles :

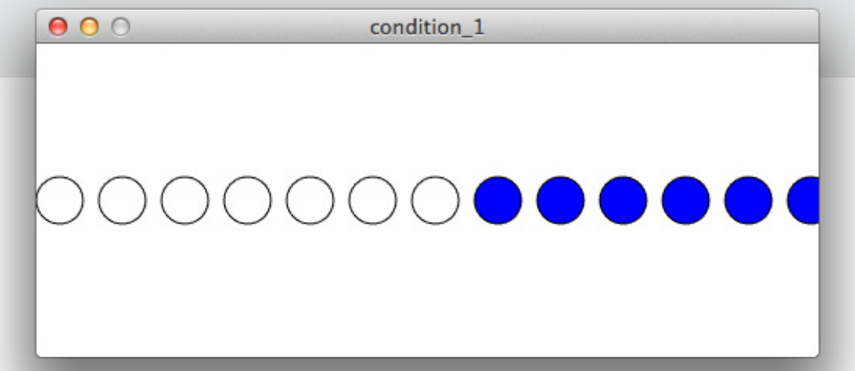
- If (si...)
- If else (si... sinon...)
- If else if (si... sinon si...)
- switch (Dans le cas où ...)

Dans un premier temps nous nous concentrerons sur les 3 premières structures. Reprenons notre première iteration et ajoutons la condition suivante :

> Si i est supérieur à la moitié de la largeur de la scène alors mes cercles sont bleu.

Il s'agit ici d'une condition simple if(). Elle se traduira par :

```
for(int i=0; i<width; i+=40)
 {
      if(i > width/2)
      {
            fill(0,0,255);
      }
  ellipse(i+15, height/2, 30, 30);
 }
```

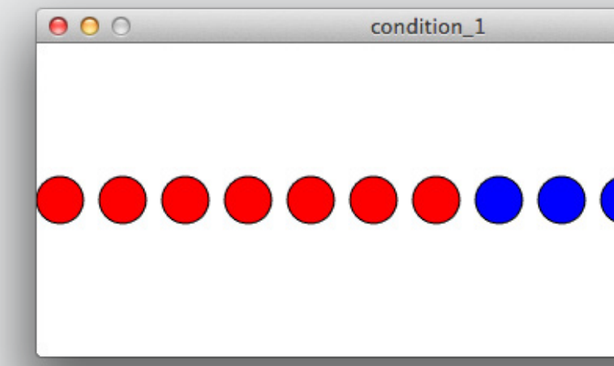


Ajoutons maintenant une second condition à notre phrase :

> Si i est supérieur à la moitié de la largeur de la scène alors mes cercles sont bleu, sinon, il sont rouge.

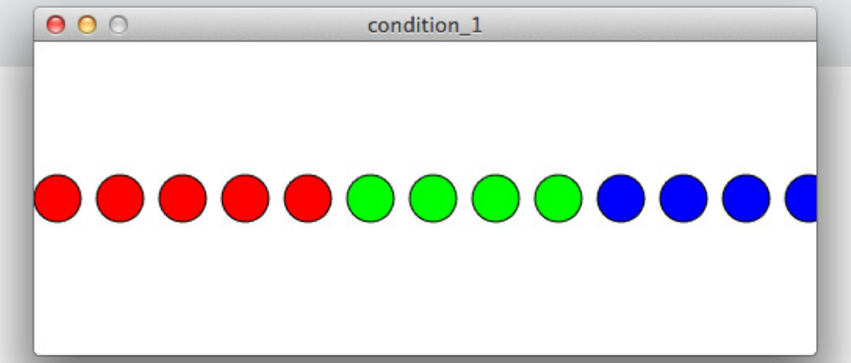
Il s'agit ici d'une condition if()...else{}. Elle se traduira de la manière suivante

```
for(int i=0; i<width; i+=40)
 {
      if(i > width/2)
      {
            fill(0,0,255);
      }
      else
      {
            fill(255,0,0);
      }
  ellipse(i+15, height/2, 30, 30);
 }
```



3 - Structure conditionnelle : construction

```
for(int i=0; i<width; i+=40)
 {
  if(i > width/3*2)
  {
   fill(0,0,255);
  }
  else if(i > width/3)
  {
   fill(0,255,0);
  }
  else
  {
   fill(255,0,0);
  }
  ellipse(i+15, height/2, 30, 30);
 }
```



Enfin complexifions un peu plus notre phrase de la manière suivante :

> **Si i est supérieur à 1/3 de la largeur de la scène alors mes cercles sont vert, sinon si i est superieur a 2/3 de la largeur de la scène alors mes cercles sont bleu, sinon il sont rouge.**

Il s'agit ici d'une condition multiple if()...else if(). Elle se traduira de la manière suivante :

3 - Structure conditionnelle : construction

Il est est possible de définir plusieurs conditionnelles de type OU ou ET à l'aide des opérateurs logique || et &&.

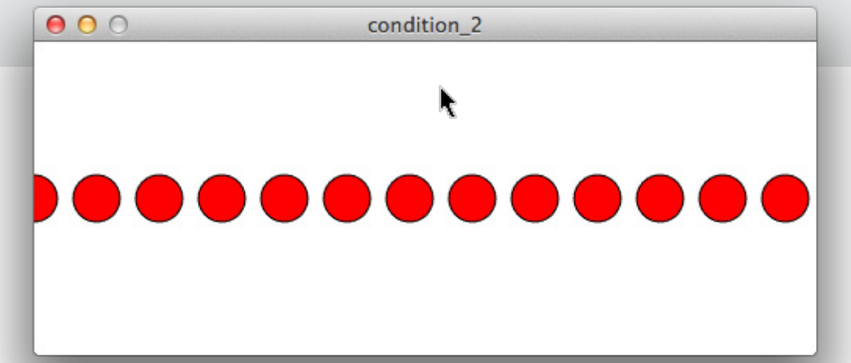
> **Ainsi nous pourrions dire : Si la position Y de la souris est supérieur à la moitié de la hateur de la scène ET que i est paire alors nos cercles sont bleu, sinon il sont rouge.**

Cela se traduirai par l'utilisation de l'opérateur logique && dans notre condition.

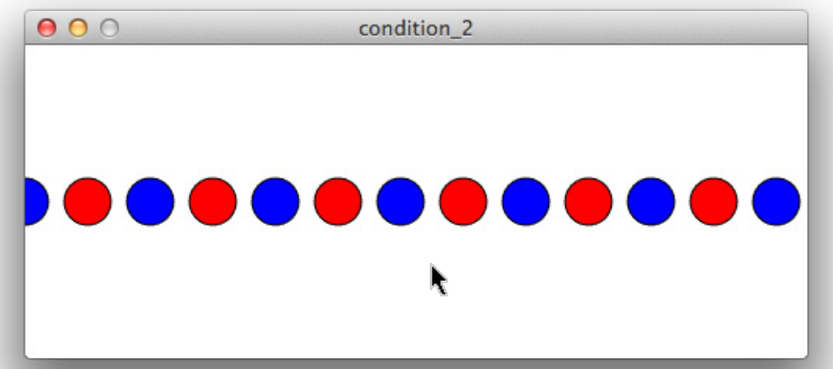
NB : Pour savoir si i est paire ou impaire nous utiliserons le modulo %. Cette opérateur permet de connaitre la valeur résiduelle d'une division, c'est à dire son reste. Or si on divise un nombre paire par 2 sont reste nulle. Cela se traduira en code par if(i%2 == 0).

```
for(int i=0; i<width; i++)
 {
   if(mouseY > height/2 && i%2 ==0)
   {
     fill(0,0,255);
   }
   else
   {
     fill(255, 0, 0);
   }
   ellipse(i*40, height/2, 30, 30);
 }
```

mouseY < height/2 &&
i%2 == 0



mouseY > height/2 &&
i%2 == 0



3 - Structure conditionnelle : construction

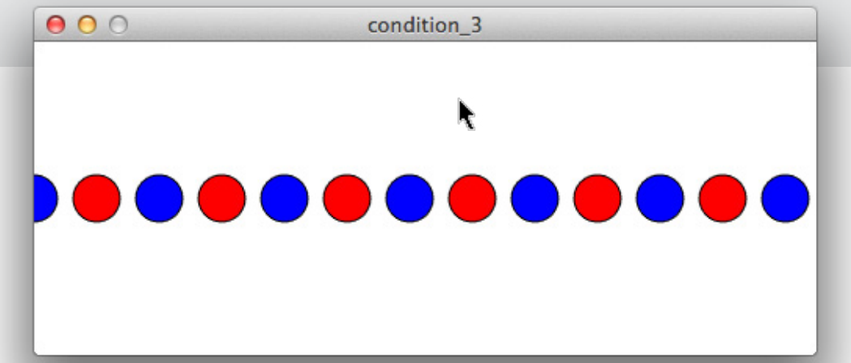
De la même manière que la double condition (&&) il est possible de réaliser une condition OU à l'aide de ||

> **Ainsi nous pourrions dire : Si la position Y de la souris est supérieur à la moitié de la hateur de la scène OU que i est paire alors nos cercles sont bleu, sinon il sont rouge.**

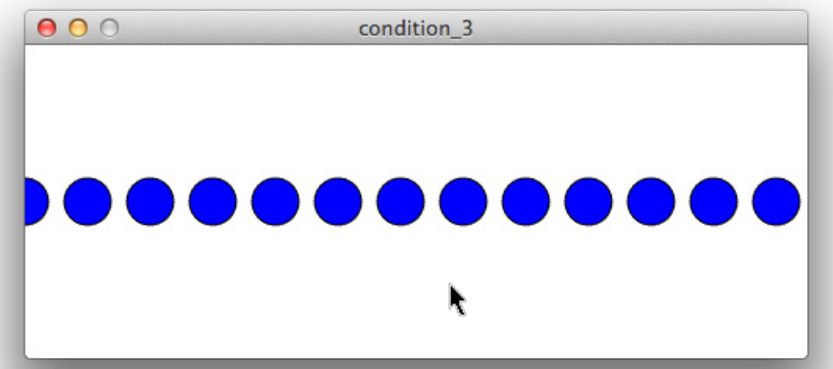
Cela se traduirai par l'utilisation de l'opérateur logique || dans notre condition.

```
for(int i=0; i<width; i++)
 {
   if(mouseY > height/2 || i%2 ==0)
   {
     fill(0,0,255);
   }
   else
   {
     fill(255, 0, 0);
   }
   ellipse(i*40, height/2, 30, 30);
 }
```

mouseY < height/2 ||
i%2 == 0



mouseY > height/2 ||
i%2 == 0



3 - Structure conditionnelle : construction

La dernière structure conditionnelle est la structure switch. Son fonctionnement est très proche de la structure if()...else if() mais celle-ci est plus pratique dans le cas où nous avons deux ou trois cas/condition à vérifier. Celle-ci fonctionne de la manière suivante :

```
int num = 1;

switch(num) {
  case 0:
    println(«Zero»);  // Does not execute
    break;
  case 1:
    println(«One»);  // Prints «One»
    break;
}
```

Elle se traduit de la manière suivante

> **Pour la variable entière i. Si celle ci est égale à 0 alors s'effectue l'action println(«Zero»), si elle est égale à 1 alors s'effectue l'action println(«Un»).**

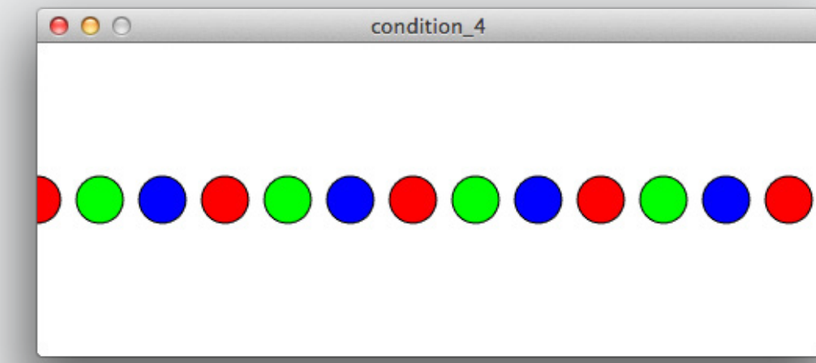
Prenons la en main de la manière suivante :

> **Pour une variable entière «pattern», si celle-ci est égale à 0 alors mes cercle seront rouge, si celle-ci est égale à 1 alors mes cercles seront vert, enfin, si celle-ci est égale à 2 alors mes cercles seront bleu.**

NB : Ici nous utiliserons de nouveau le modulo. En effet dans le cas d'une boucle de nombreux calculs à l'aide de modulo permettent de créer des suites.Ici nous définirons patterne à l'aide de i%3 = 0,1,2,0,1,2,0,1,2...

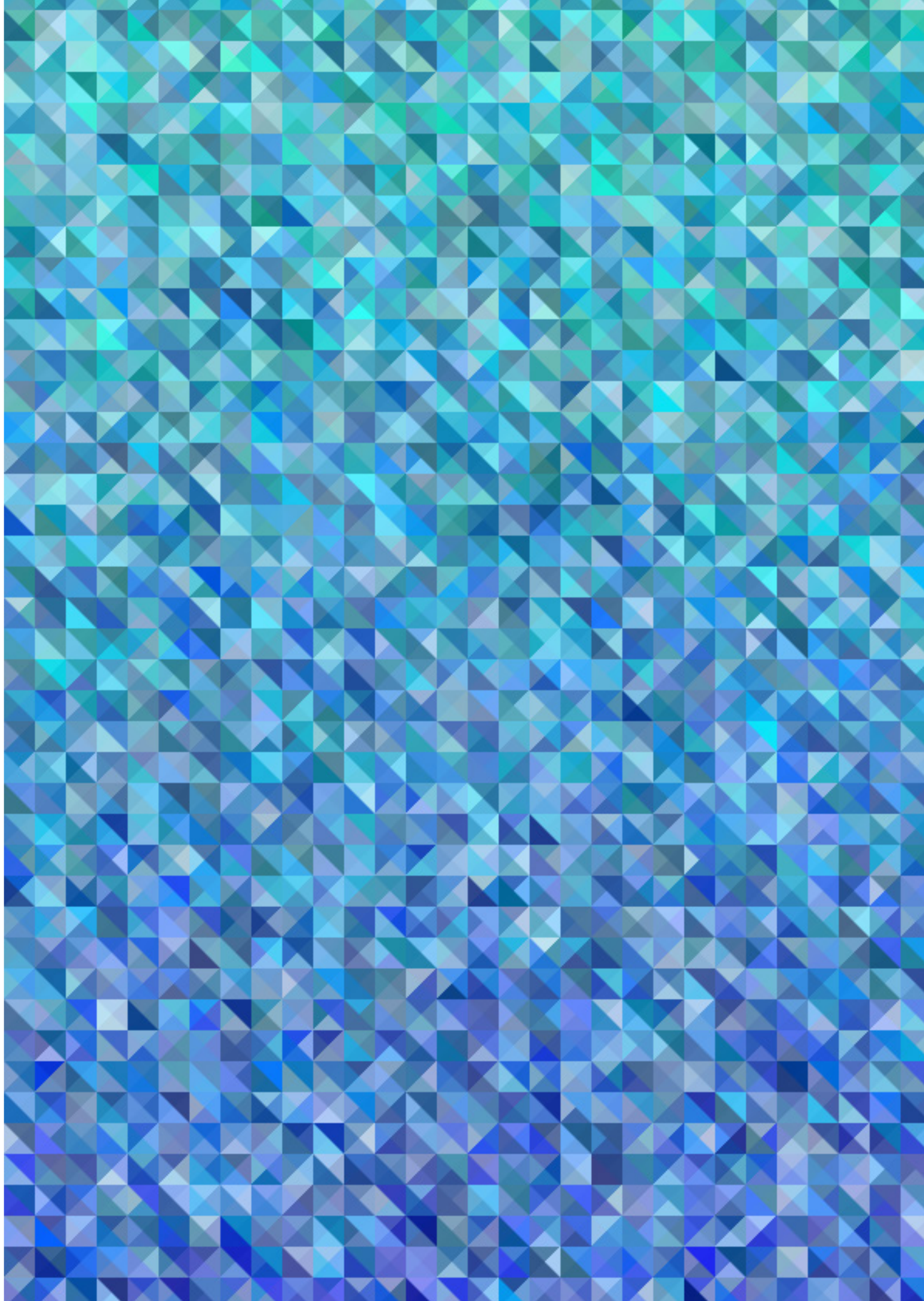
```
for (int i=0; i<width; i++)
 {
  int pattern = i%3;

  switch (pattern)
  {
  case 0:
    fill(255, 0, 0);
    break;
  case 1:
    fill(0, 255, 0);
    break;
  case 2:
    fill(0, 0, 255);
    break;
  }
  ellipse(i*40, height/2, 30, 30);
 }
```

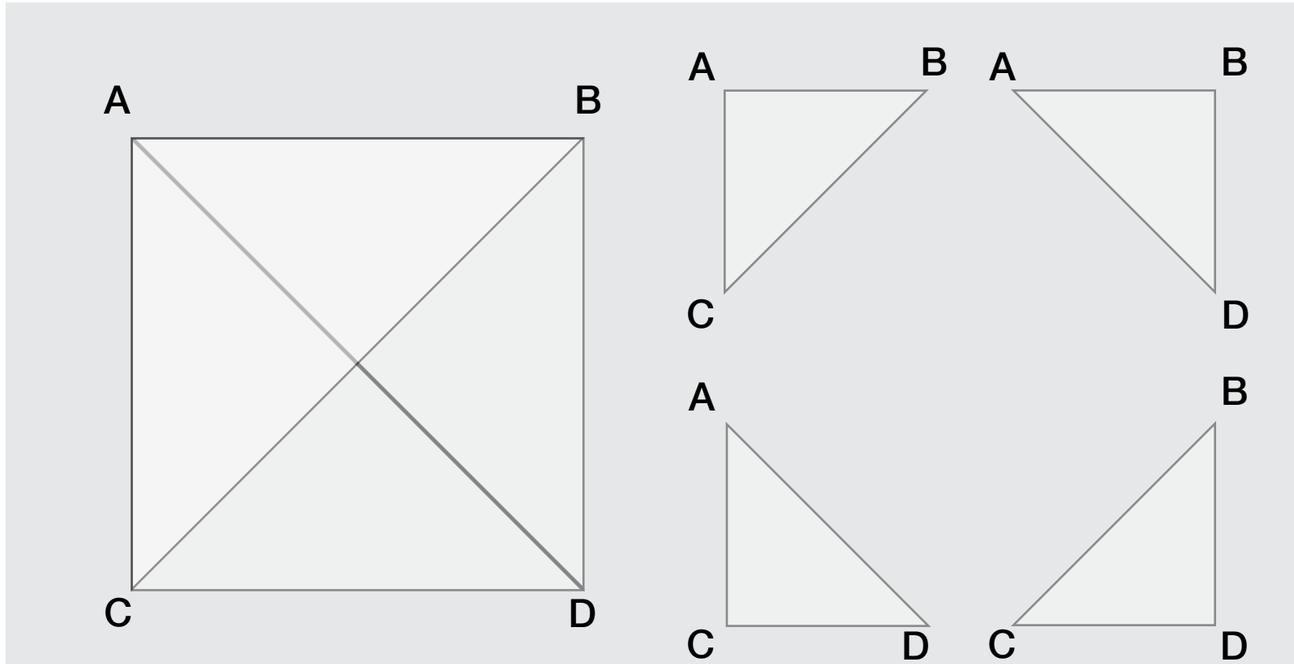


4 - Un exemple de motif utilisant les structures conditionnelles et itératives

Dans cette exemple nous verrons comment utiliser les iterations afin de réaliser une motif composé de triangles.



4 - Un exemple de motif utilisant les structures conditionnelles et itératives



Nous allons dans une premier temps définir la composition de notre motif. Ici nous souhaitons obtenir un motif à de superposition de triangles de teinte proche et ayant suffisament de transparence pour laisser apparaitre leur superposition.

Nous aurons aussi besoins de créer une grille à l'aide d'une double boucle for et d'en définir la résolution.

Enfin nous utiliserons la méthode colorMode() pour travailler en teinte saturation et luminosité.

Dans un premier temps nous allons devoir définir les variables dont nous aurons besoins :

- Position x y des sommest A, B, C, D
- Teinte et increment de teinte pour ne pas uniformiser les couleurs
- Resolution de la grille
- Random Seed (permettant de définir une constante à une méthode aléatoire)

4 - Un exemple de motif utilisant les structures conditionnelles et itératives

```
int sWidth = 400;
int sHeight = sWidth*2;

int xA, yA, xB, yB, xC, yC, xD, yD;
float hue;
int Spacing=20;
int colorInc = 20;
int actualRandomSeed = 1000;
```

Nous définissons ensuite le constructeur de notre skecth

```
void setup()
{
  size(sWidth, sHeight, P2D);
  colorMode(HSB, 360, 100, 100, 100);
}
```

Enfin nous définissons notre boucle ***draw().***
Nous utiliserons la methode ***randomSeed()*** afin de définir une constante pour tout les nombre aléatoire que nous générerons

```
void draw()
{
  background(0, 0, 100);
  randomSeed(actualRandomSeed);
}
```

Nous créons ensuite notre double boucle for afin d'établir notre grille de motif de résolution ***«Spacing»***

```
 for (int gridX = 0; gridX<width/Spacing; gridX ++)
  {
    for (int gridY = 0; gridY<height/Spacing; gridY ++)
    {
    }
}
```

4 - Un exemple de motif utilisant les structures conditionnelles et itératives

Enfin dans notre double boucle *for()* nous définissons les position de nos sommets où :

```
xA = gridX*Spacing;
yA = gridY*Spacing;
xB = xA+Spacing;
yB = yA;
xC = xA;
yC = yA+Spacing;
xD = xA+Spacing;
yD = yA+Spacing;
```

Afin de créer un dégradé de couleur nous utiliserons la méthode *map()* afin de définir la teinte gloable de nos triangle en fonction de leur position sur l'axe Y. La méthode *map()* permet de réaliser une règle de trois afin de mapper une valeur d'un rang de valeurs à un autre rang de valeurs

```
xA = gridX*Spacing;
yA = gridY*Spacing;

xB = xA+Spacing;
yB = yA;

xC = xA;
yC = yA+Spacing;

xD = xA+Spacing;
yD = yA+Spacing;


hue = map(yA, 0, height, 180, 230);
```

4 - Un exemple de motif utilisant les structures conditionnelles et itératives

Enfin, notre dernière étape consistera à déssiner nos triangle. Afin d'obtenir des triangles de teinte et d'opacité différentes nous utiliserons la méthode *random()* afin de celle-ci soit défini de manière aléatoire.

```
noStroke();

    //ABC
    fill(random(hue-colorInc, hue+colorInc), random(50, 100),
random(50, 100), random(20, 100));
    triangle(xA, yA, xB, yB, xC, yC);

    //BCD
    fill(random(hue-colorInc, hue+colorInc), random(50, 100),
random(50, 100), random(20, 100));
    triangle(xB, yB, xC, yC, xD, yD);

    //ABD
    fill(random(hue-colorInc, hue+colorInc), random(50, 100),
random(50, 100), random(20, 100));
    triangle(xA, yA, xB, yB, xD, yD);

    //ADC
    fill(random(hue-colorInc, hue+colorInc), random(50, 100),
random(50, 100), random(20, 100));
    triangle(xA, yA, xD, yD, xC, yC);
```

5 - Exercice : Les grilles et répétitions de motifs

À l'aide de vos connaissances acquises en cours, réaliser une série de 6 visuels basés sur des grilles et répétitions de motifs.

La série devra être évolutive et chaque visuel pourra être reconnu comme étant une évolution du précédent.

La série se présenter sous forme de poster dont la nomenclature sera fourni et devra être respecté.

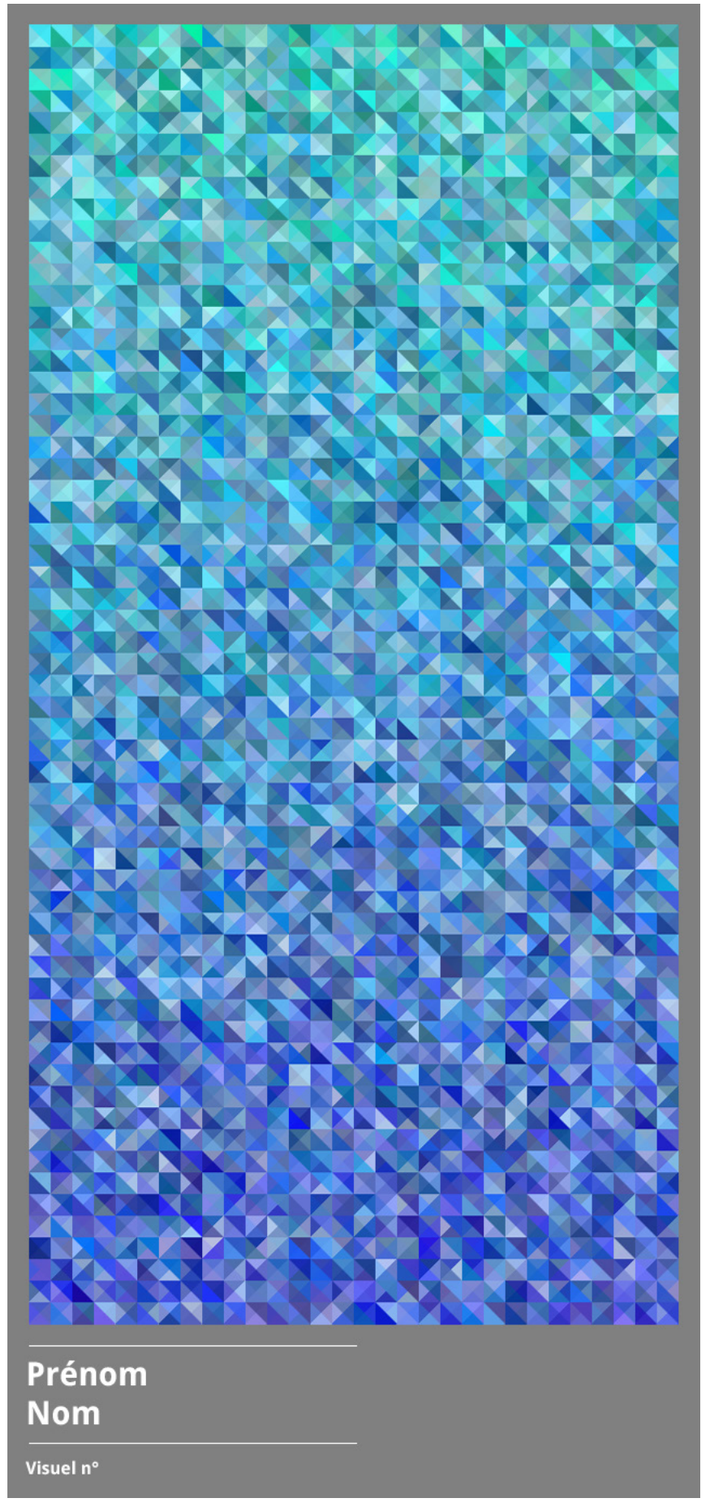
<u>Contraintes :</u>

- **Format du Sketch : 600*1200**
- **Format du poster : 640*1380**

<u>Rendu :</u>

- **Images au format Jpg**
- **Sketch processing**

L'ensemble de fichiers neceassaire àa la réalisation du sujet sont téléchargeable via GitHub à l'adresse suivante : *https://github.com/alexr4/eartsup-course*



Contact

Alexandre Rivaux
Visual Designer & Partner Bonjour, interactive Lab

www.bonjour-lab.com
arivaux@gmail.com