PROCESSING

Ordonner un processus aléatoire

1. Librairie Géomérative

2. La classe Balle

Projet personnel

Elisa Raffy

1 Présentation du projet

/ Demande

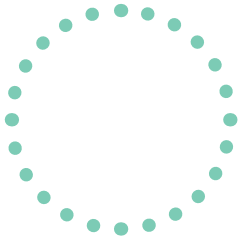
A partir du sigle *Design 4*, créer un logo génératif.

/ Mon concept

A partir du rond qui symbôlise l'unité entre les quatres DSAA, créer le chiffre 4 avec des balles.

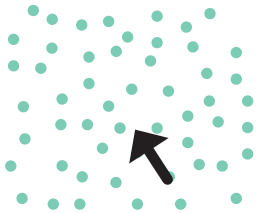
Le rond (balle)

Symbole d'unité entre les quatres DSAA



Balle + cercle

Alliance entre les sections



Souri + mouvement

dispertion des balles

Formation du 4

1 Déroulement / étapes

- On ouvre la **bibliothèque Géomérative** avec ces fonctions.
(la bibliothèque permet de former le **chiffre 4** avec des balles).

- Le tableau **Arraylist** créer la classe «ensembleBalles»
(classe qui regroupera toutes les balles du 4).

- Pour inscrire le texte «4» on met la fonction **string**,
qui créer une chaîne de caractères.

→ **Le setup**

- Lancer le **setup**.
(cela permet d'ouvrir le code et dans notre cas de trouver les points vectoriel de la police).

- On rentre la taille de la fenêtre (***size)***, la couleur du fond (***background***) et l'anti pixellisation (***smooth***).

- Ensuite, la fonction **RG.int** initialise les objets de la bibliothèque.

- pour incorporer une police on utilise la fonction **RFont**.
La police est chargé à partir de la librairie. Pour la récupérer il suffis d'aller dans son dossier de police, et de la faire glisser sur le sketch.

- A partir de cette police, on détermine une taille et une position
(«FreeSans.ttf», 250, CENTER).

- Ensuite pour écrire le mot «DESIGN», on charge une police a partir du sketch. On la copie dans le dossier DATA, et ensuite on la copie dans le programme.
La fonction **Loadfont** définis une police spécifique, qui appartient au dossier du sketch. C'est l'inverse d'une police **Newfont,** qui elle est universel (commune à tous).

En placant la fonction **fill**, on détermine les couleurs des balles.
La fonction **noStroke**, détermine si oui ou non on souhaite un contour.

- On créer le premier objet : Objet de regroupement par la fonction **my Group.**
- Pour trouver les points vectoriel de la police, on créer une **boucle** entre 0 et a taille maxi. La taille maxi est déterminé par la fonction **length.**

```
for (int i = 0; i < myPoints.length; i++
```

- la fonction nous indique qu'elle a trouvé 195 points.

```
ensembleBalles.add(new Balle(myPoints[i].x + width /2,
myPoints[i].y + height / 2));
```

→ **Le draw**

- Pour changer la couleur, taille et texte «DESIGN», on place ces éléments dans le **draw**.

- Pour que ces changements ne s'applique pas au balles, on utilise deux fonctions : **pushStyle** et **popStyle**.

→ **Création de la classe Balle**

- Pour faire varier les mouvements et positions des balle, on créer une classe.

- On utilise la fonction **PVector**, composé de trois objets : position, origine et destination. Ces objets vont nous permettre de déterminer une destination au hasard pour les balles.

- La fonction **PVector**, permet aussi de places des formes sans répéter la fonction.

- Pour interpoller des formes entre elles, on utilise la fonction **lerp**.

- Enfin la fonction **ellipse** déterminera le diamètre de chaque balles.

2 Librairie Géomérative

Pour former le chiffre 4 avec des balles,
il faut faire appel à la bibliothèque Géomérative.

La bibliothèque est composé de plusieurs fonctions.
Pour ce programme, on utilise les fonctions :

Rpoint, Rgroupe, Rcommande, Rfont...

Début du programme

```
import geomerative.*; //ouverture de la bibliothèque

RFont font; //ouverture de la fonction Rfont
PFont maTypo;

ArrayList<Balle> ensembleBalles; //ArrayList c'est un tableau
String texte = "4"; //fonction String : créer une chaîne de carcatères
```

FONCTIONS

Font : induit que la bibliothèque a créer un fichier typo
Rfont : ouvrir une typo à partir de la bibliothèque

ArrayList : c'est une classe de processing, une forme de stockage des données dynamiques. (Une forme de tableau dont on peu enlever et ajouter des éléments).
String : c'est pour créer une chaine de caractères. Ici le chiffre «4»

Fonctions dans le setup

Le **setup** permet de trouver tout les points vectoriel du caractère typographique, et de les figés. Le setup n'est pas «animé», c'est dans celui-ci que l'on utilise les fonctions de la bibliothèque Géomarative

setup

Les « **{** » ouvrent et ferment la classe.

- Ici le setup trouve les points vectoriel de la police et les regroupe ensemble.

```
void setup() {

  size(800, 800);
  background(255);
  smooth();
  RG.init(this);


  font = new RFont("FreeSans.ttf", 500, CENTER);

  maTypo = loadFont("HelveticaRoundedLT-Bold-48.vlw");

  fill(0,940,164);
  noStroke();

  RCommand.setSegmentLength(5);
  RCommand.setSegmentator(RCommand.UNIFORMLENGTH);

  RGroup myGroup = font.toGroup(texte);
  myGroup = myGroup.toPolygonGroup();

  RPoint[] myPoints = myGroup.getPoints();
  ensembleBalles = new ArrayList<Balle>();
  for (int i = 0; i < myPoints.length; i++) {

    ensembleBalles.add(new Balle(myPoints[i].x + width /2, myPoints[i].y + height / 2));
  }
}
```

Fonctions dans le setup

charger la typo par rapport a la librairie //typo pour le 4 + taille et placement

- taille
- fond (arrière plan)
- anti pixelisation

Initialise l'objet constructeur de la bibliothèque

Chargement de la typographie par rapport à la librairire

Changer la police de «DESIGN»

- remplir avec une couleur
- pas de couleur autour des balles

Créer un objet de regroupement

Il créer un assemblage avec la **fonction myGroup**

- Trouver les point d'encrage de la police. **myPoints** est une variable
- **Arraylist** : liste des balles sous forme de tableau «*ensembleBalles*»

Tableau «*ensembleBalles*» qui correspond a 195pts

```
void setup() {

  size(800, 800);
  background(255);
  smooth();
  RG.init(this);


  font = new RFont("FreeSans.ttf", 500, CENTER);

  maTypo = loadFont("HelveticaRoundedLT-Bold-48.vlw");

  fill(0,940,164);
  noStroke();

  RCommand.setSegmentLength(5);
  RCommand.setSegmentator(RCommand.UNIFORMLENGTH);

  RGroup myGroup = font.toGroup(texte);
  myGroup = myGroup.toPolygonGroup();

  RPoint[] myPoints = myGroup.getPoints();
  ensembleBalles = new ArrayList<Balle>();
  for (int i = 0; i < myPoints.length; i++) {

    ensembleBalles.add(new Balle(myPoints[i].x + width /2, myPoints[i].y + height / 2));
  }
}
```

Nom , taille et position dans le background

loadFont : c'est une police spécifique qui appartient au dossier du sketch.

Création d'une boucle.
Il cherche tout les points d'encrage de la forme vectoriel (police) entre 0 et la taille maxi.

length : détermine la taille maxi

Fonctions dans le draw

La fonction draw vas dans ce sketch me permettre de faire une boucle et d'animer l'explosion de balles.

Fonction float :
variable décimale.
Coordonées de la classe Balle en rapport avec myPoints.

Réaliser une action pour l'ensemble des balles

Fonction pushStyle et popStyle

tout ce qu'on met entre ces deux crochets ne s'applique pas au reste du sketch. Dans mon cas, elle ne change pas la couleur de mes balles en noir.

```
void draw() {
  background(255);
  for (Balle b : ensembleBalles) {
    b.miseAJour((float)mouseX / width);
    b.affichage();
  }
  pushStyle();
  fill(115);
  textFont(maTypo, 50);
  //textSize(50);
  text("DESIGN ", 150, 300);
  popStyle();
}
```

3 Classe Balle

La classe Balle est la partie qui détermine la position, l'origine et la destination des balles. Pour cela on utilise la fonction **PVector.**

PVector :
Cette fonction est utilisé pour décrire une position , de vitesse ou d'accélération. Elle permet aussi de ne pas permet de ne pas réecrire **x** et **y** et de **combiner** les deux coordonées.

détermine l'origine, la destination et la position des balles

appel la classe Balle

Créer des destination au hasard pour la balle

lerp :
créer une transission entre le points d'origine et le point d'entrée

détermine le diamètre et la position de l'llipse de chaque balles

```
class Balle {
  PVector origine, destination, position;

  Balle(float x, float y) {


    origine = new PVector(x, y);
    position = origine.get();
    destination = new PVector((int)random(0, width), (int)random(0, height));

  }

  void miseAJour(float parametre) {
    position = PVector.lerp(origine, destination, parametre);

  }

  void affichage() {
    ellipse(position.x, position.y, 10, 10);
  }
}
```

3 Classe Balle

Le 4 avec ces balles est situé sur un axe X.
Celui ci explose en fonction du mouvement du pointeur.
Plus on s'éloigne du point 0, plus le 4 est reconnaissable. A l'inverse, plus on s'éloigne et plus il explose.