# Projet compléments programmation orientée objet

## Ensembles de Julia

Projet réalise par le binome 25 composé de :

- BENMEBAREK Rafik
- DOUAH Ilies

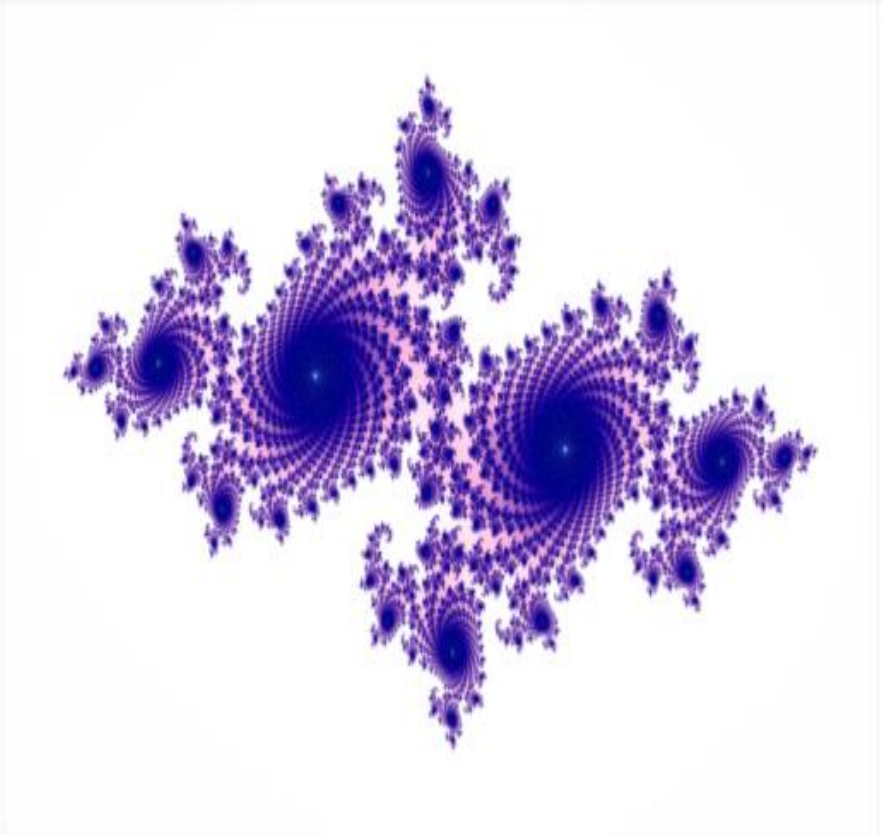


*Figure 1 : Un ensemble de julia (source : Wikipedia)*

# 1. Fonctionnalités

Tout d'abord, le projet est exécutable en mode IG (Interface Graphique) tout comme en mode terminale, donc pour chaque fonctionnalité il y aura une explication pour chacun de ces deux modes.

Dans cette partie, nous allons faire une description des fonctionnalités implémentés dans le projet.

## a. Génération de fractales :

Dans le cadre de ce projet, nous avons travaillé sur la génération de fractales, notamment les ensembles de Julia et Mandelbrot. Il est aussi possible de générer n'importe quel autre fractale en tant que polynôme de degré quelconque.

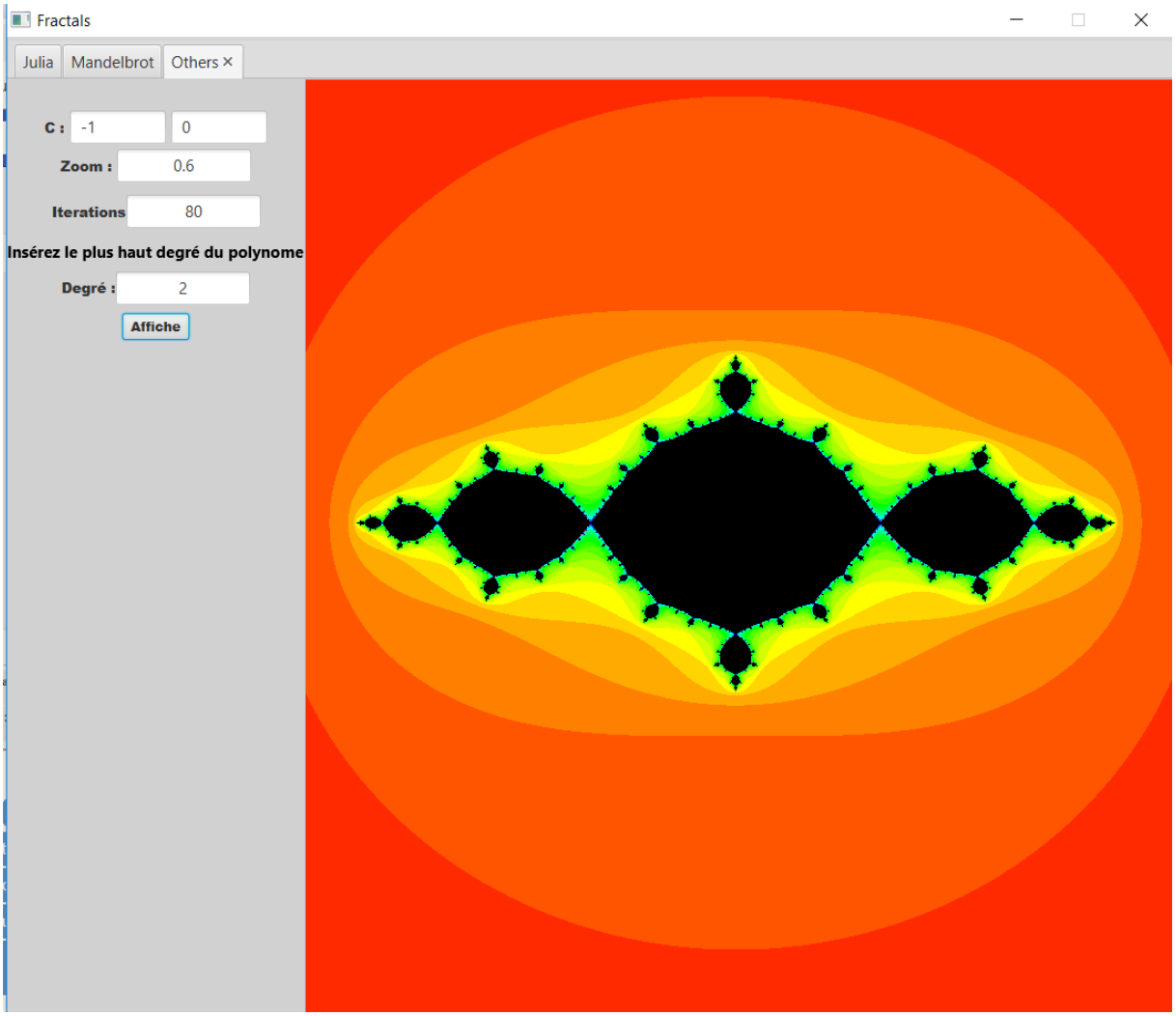



*Figure 2 : Interface graphique*

Dans la figure 2, nous sommes dans le cas d'une génération d'une fractale ayant un degré quelconque (c'est-à-dire, qui n'est pas un ensemble de Julia ou de

Mandelbrot). Pour procéder à l'affichage il suffit de fournir le nombre complexe C, le nombre d'itérations maximum pour s'assurer de la convergence.

La génération d'ensemble de Julia et de Mandelbrot est aussi incluse, la seule différence avec la capture d'écran, est l'absence de champs « degré ».

### b. Comment se fait la génération ?

Dans les premières phases de l'implémentation, la génération se faisait grâce à l'utilisation du PixelWriter présent dans la classe GraphicsContext. Mais le problème était que la génération prenais trop de temps (7 secondes pour 200 itérations).
Nous avons ensuite essayé d'utiliser 4 threads auxiliaire, et chaque thread s'occupe de dessiner un quart de l'image.
Le problème auquel nous avons fait face, est que le PixelWriter n'était pas threadsafe, donc il n'y avait aucune amélioration au niveau du temps d'affichage.
Nous avons adopté alors une autre méthode, en créant une classe FastDrawer qui implémente l'interface Runnable, sa méthode run s'occupe de déssiner la fractale sur une BufferedImage pixel par pixel grâce à sa méthode setRGB() qui est totalement threadsafe.

```
float brightness = itr < fractal.maxItr ? 1f : 0;
float hue = FractalSet.affectColor(itr);
double[] rgb = Utils.HSBtoRGB(hue, 1.0f, brightness);
image.setRGB(x, y, new Color(
        (int) (rgb[0] * 255),
        (int) (rgb[1] * 255),
        (int) (rgb[2] * 255)
        ).getRGB()
        );
```

Nous avons remarqué une nette amélioration en terme de rapidité, car nous sommes passés de 7 secondes (sans thread) à 3 secondes !

Après la création de la BufferedImage, il y a deux utilisations possibles :

- En IG mode : l'image est convertie en image FX (car BufferedImage est puis dessinée dans le GraphicsContext comme le montre le code ci-dessous

```
public void showGraph(TriFractalFunction f){
    BufferedImage image = paint(f);
    Image img = SwingFXUtils.toFXImage(image, null);
    gc.drawImage(img, 0, 0);
}
```

- En mode terminale : L'image est directement sauvegardée en faisant l'appel en tant que .png dans le répertoire racine du projet.

## 2. Comment compiler/exécuter ?

Compilation :

1. Pour la compilation, les étapes sont simples, il suffit de se diriger vers le dossier du projet, et y ouvrir l'invite de commande (cmd).
2. Taper la commande qui se trouve dans la capture d'écran ci-dessous

```
C:\Users\Rafik\workspace\Julia\src>javac Main.java
```

Exécution

- Pour l'exécution en mode IG : un simple « java Main » dans l'invite de commande suffira.

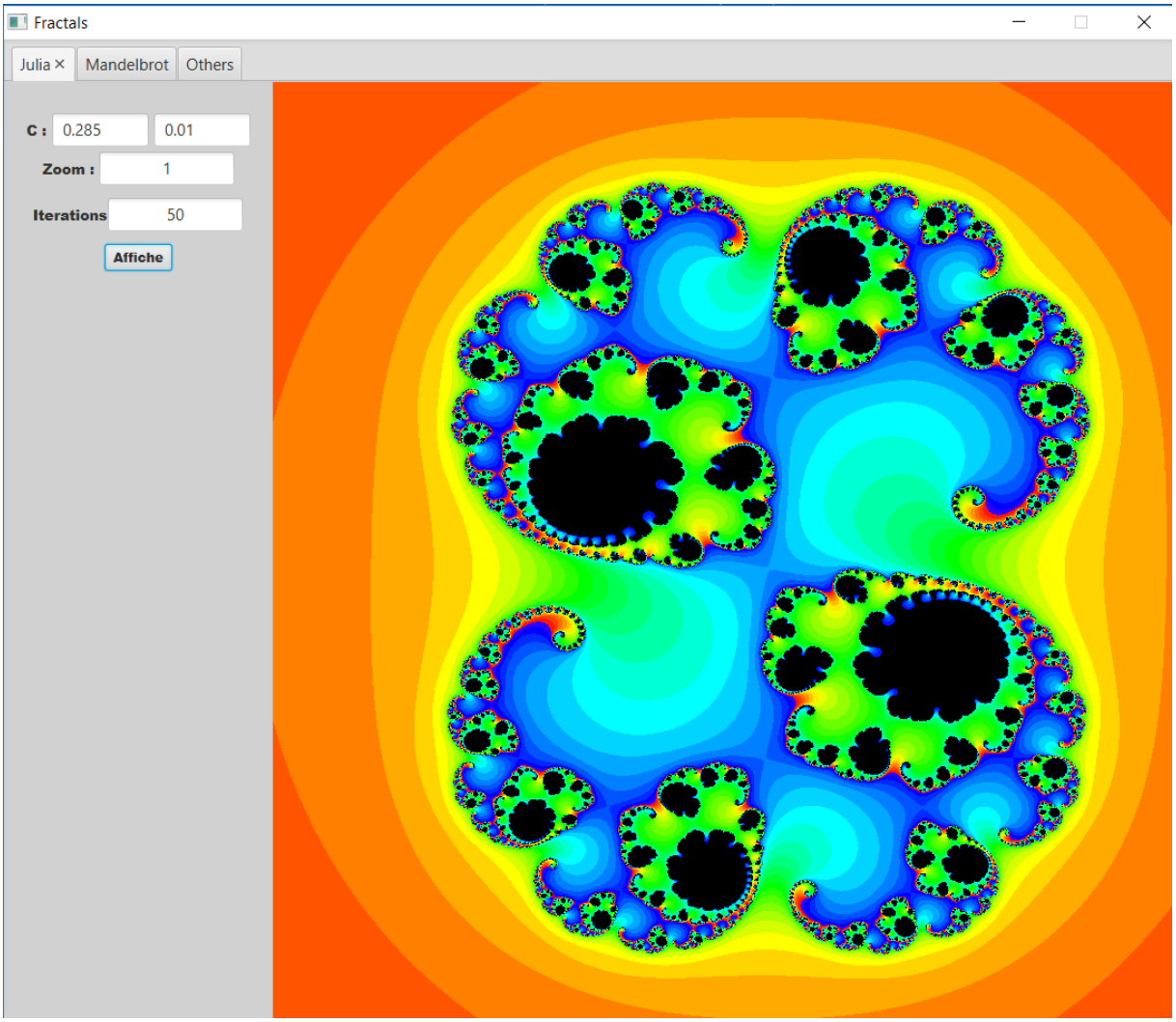
```
C:\Users\Rafik\workspace\Julia\src>java Main
janv. 04, 2019 6:04:00 PM javafx.fxml.FXMLLoader$ValueElement processValue
WARNING: Loading FXML document with JavaFX API of version 10.0.1 by JavaFX runtime of version 8.0.191
```

- Pour l'exécution en mode terminale : on ajoutera un argument a la commande d'exécution qui est « cli »

```
C:\Users\Rafik\workspace\Julia\src>java Main cli
*****************************FRACTALS*****************************
1.Ensemble de Julia
2.Ensemble de Mandelbrot
3.Autre
Quel ensemble souhaitez vous générer? _
```

## 3. Mode d'emploi ?

Il n'y a pas de mode d'emploi, car l'interface graphique est très simple et évidente pour quelqu'un qui en sait un minimum sur les fractales.




Et il en est de même pour le mode terminale, car le programme s'occupe de vous guider jusqu'à la génération de votre fractale, et finit par vous donner le chemin où elle a été sauvegardée.

```
C:\Users\Rafik\workspace\Julia\src>java Main cli
*******************************FRACTALS*******************************
1.Ensemble de Julia
2.Ensemble de Mandelbrot
3.Autre
Quel ensemble souhaitez vous générer? 1
Vous avez choisis l'ensemble de Julia
Entrez le nombre d'itération : 50
Entrez le zoom : 1
Entrez la partie réel du C : 0,285
Entrez la partie imaginaire du C : 0,01
Génération en cours...
Fractale générée dans : C:\Users\Rafik\workspace\Julia\src\Julia2.png

C:\Users\Rafik\workspace\Julia\src>
```

Petite remarque : Java reconnait les nombres réels en suivant un certain pattern, et ce pattern demande la présence d'une virgule « , » et non un point « . » comme on aurait l'habitude de l'écrire.