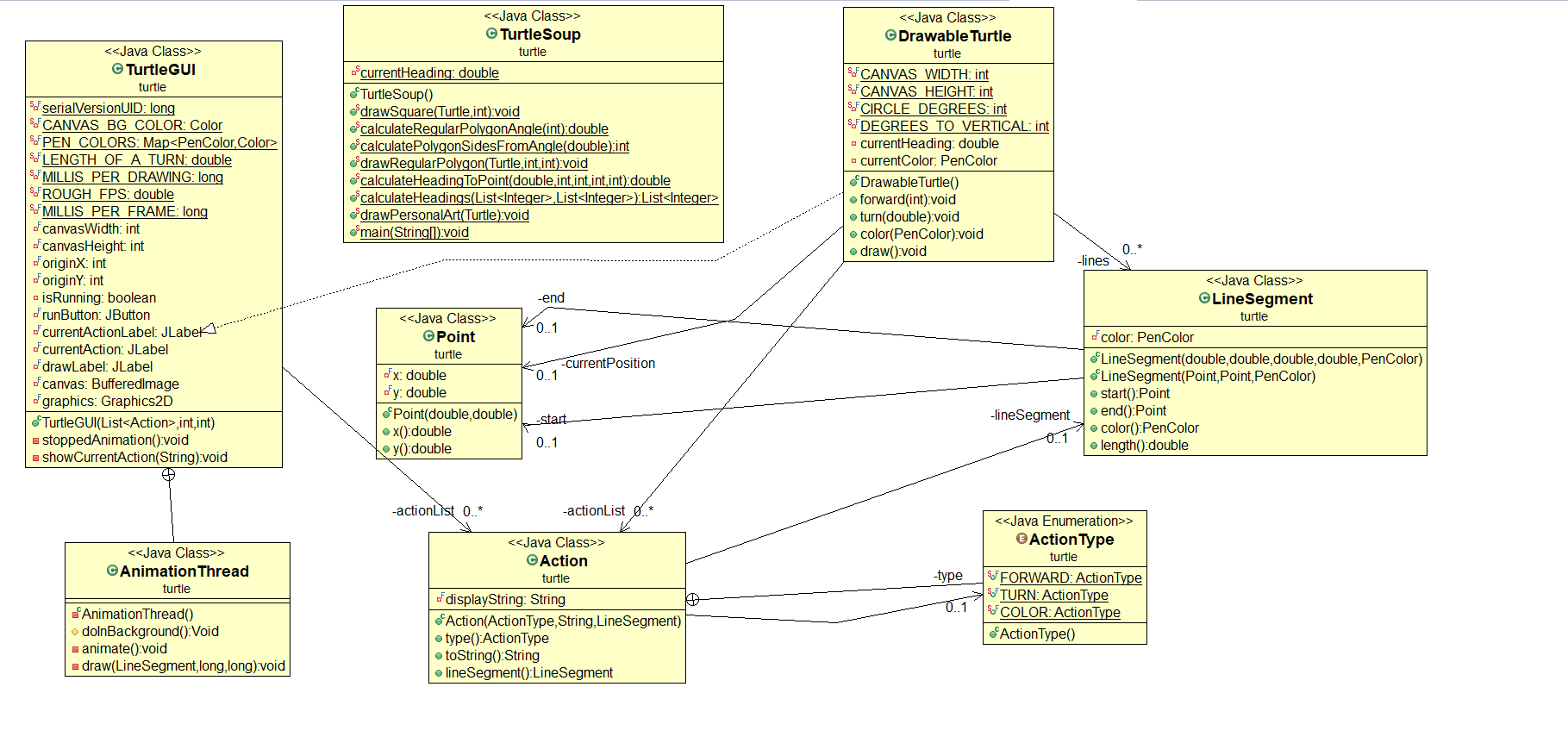
# **RAPPORT**

# **PROJET 1 DU COURS DE DESIGN PATTERN 2019**

Ce présent document représente le rapport de mon premier projet du cours de design pattern de M. Bousso. Il met l’accès sur l’explication et la description des différentes captures d’écran sur le rendu des polygones à travers du code java.

L’ensemble des tests sont effectués dans la classe TurtleSoup.java.

# Diagramme de classe :



# Capture 1 :

Une image contenant capture d’écran

Description générée automatiquement

Cette capture contient un carré avec des couleurs différentes sur les côtés. Elle est générée par la méthode statique suivante :

**public** **static** **void** drawSquare(Turtle turtle, **int** sideLength) {

//throw new RuntimeException("implement me!");

turtle.forward(sideLength);

turtle.turn(90.0);

turle.color(PenColor.***RED***);

turtle.forward(sideLength);

turtle.turn(90.0);

turtle.color(PenColor.***CYAN***);

turtle.forward(sideLength);

turtle.turn(90.0);

turtle.color(PenColor.***GREEN***);

turtle.forward(sideLength);

turtle.turn(90.0);

turtle.color(PenColor.***ORANGE***);

}

# Capture 2 :

Une image contenant capture d’écran

Description générée automatiquement

La deuxième capture représente, en fait, un polygone régulier ayant six côtés.

Pour calculer la somme de tous les angles du polygone, on établit la relation : (n - 2) \* 180 où n étant le nombre de côtés.

Voici le code source qui permet d’afficher la figure :

**public** **static** **double** calculateRegularPolygonAngle(**int** sides) {

//throw new RuntimeException("implement me!");

**double** sumAngle = (sides - 2) \* 180;

**return** sumAngle;

}

**public** **static** **void** drawRegularPolygon(Turtle turtle, **int** sides, **int** sideLength) {

//throw new RuntimeException("implement me!");

**int** sumAngle = (**int**) *calculateRegularPolygonAngle*(sides);

**int** oneAngleSide = sumAngle / sides;

**int** turnAngle = oneAngleSide / 2;

System.***out***.println("Somme des angles : " + sumAngle);

System.***out***.println("Angle pour un coté : " + oneAngleSide);

turtle.forward(sideLength);

turtle.turn(turnAngle);

turtle.color(PenColor.***RED***);

turtle.forward(sideLength);

turtle.turn(turnAngle);

turtle.color(PenColor.***BLACK***);

turtle.forward(sideLength);

turtle.turn(turnAngle);

turtle.color(PenColor.***CYAN***);

turtle.forward(sideLength);

turtle.turn(turnAngle);

turtle.color(PenColor.***ORANGE***);

turtle.forward(sideLength);

turtle.turn(turnAngle);

turtle.color(PenColor.***YELLOW***);

turtle.forward(sideLength);

turtle.turn(turnAngle);

turtle.color(PenColor.***RED***);

}

# Capture 3 :

Une image contenant capture d’écran

Description générée automatiquement

Comme demandé dans l’exercice, la troisième figure est mon œuvre issu d’inspiration personnel. Elle contient plusieurs polygones qui se coupent.

**public** **static** **void** drawPersonalArt(Turtle turtle) {

//throw new RuntimeException("implement me!");

turtle.forward(100);

turtle.turn(65.35);

turtle.color(PenColor.***GREEN***);

turtle.forward(100);

turtle.turn(65.35);

turtle.color(PenColor.***BLUE***);

turtle.forward(100);

turtle.turn(65.35);

turtle.color(PenColor.***CYAN***);

turtle.forward(100);

turtle.turn(65.35);

turtle.color(PenColor.***GREEN***);

turtle.forward(100);

turtle.turn(65.35);

turtle.color(PenColor.***RED***);

turtle.forward(100);

turtle.turn(65.35);

turtle.color(PenColor.***BLACK***);

turtle.forward(100);

turtle.turn(65.35);

turtle.color(PenColor.***ORANGE***);

turtle.forward(100);

turtle.turn(65.35);

turtle.color(PenColor.***RED***);

turtle.forward(100);

turtle.turn(65.35);

turtle.color(PenColor.***GRAY***);

turtle.forward(100);

turtle.turn(65.35);

turtle.color(PenColor.***ORANGE***);

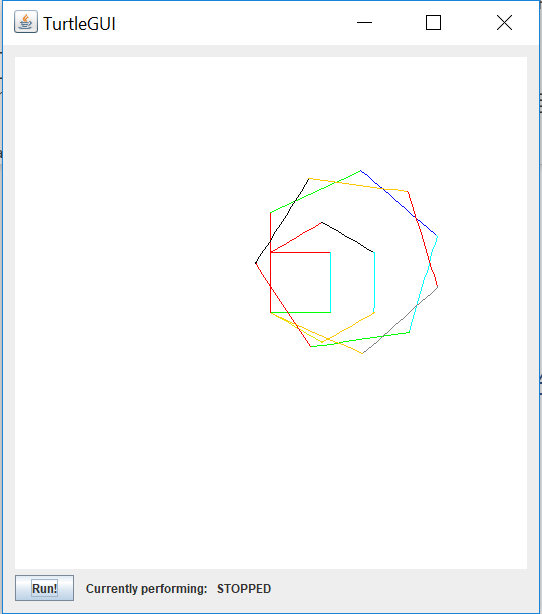
turtle.forward(100);

turtle.turn(65.35);

turtle.color(PenColor.***CYAN***);

}

# Capture 4 :



Concernant la quatrième capture, c’est une combinaison du carré, du polygone et du polygone personnel.

*drawSquare*(turtle, 60);

*drawRegularPolygon*(turtle, 6, 60);

*drawPersonalArt*(turtle);

// draw the window

turtle.draw();

# Conclusion:

Ce projet nous a permis d’appréhender le design pattern et de mettre à jour nos connaissances sur la programmation orientée objet en particulier Java.