2019

Sid Nassym Brahim Zitoun Alpha Marouana Diallo

20/01/2019

Gestion de formes géométriques



Table des matières

[1.Sujet 2](#_Toc535578322)

[2. Forme géométrique 2](#_Toc535578323)

[2.1 transformations géométriques 3](#_Toc535578324)

[2.2 Polygone 3](#_Toc535578325)

[2.3 String 3](#_Toc535578326)

[3. Dessiner une forme 3](#_Toc535578327)

[3.1 Dessiner Java 3](#_Toc535578328)

[3.2 Dessiner SFML : 5](#_Toc535578329)

[4. Sauvegarde/Chargement de forme 5](#_Toc535578330)

[4.1 Design Pattern Visitor 5](#_Toc535578331)

[4.2 Design Pattern Chain Of Responsibility 6](#_Toc535578332)

[4.3. Explode 9](#_Toc535578333)

# 1.Sujet

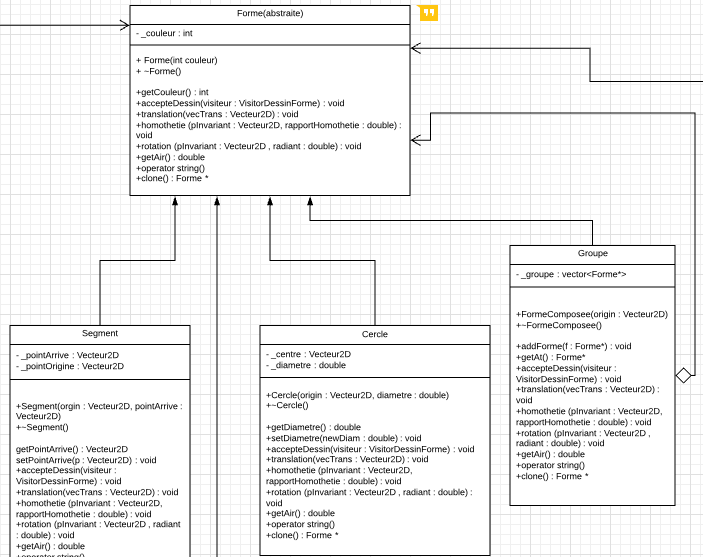
Application distribuée (client C++/serveur JAVA) permettant de gérer des formes géométriques en 2D. L'application permet de construire des formes, d'appliquer quelques transformations géométriques sur celles-ci, de les dessiner et enfin, de les sauvegarder/charger sur un fichier disque.

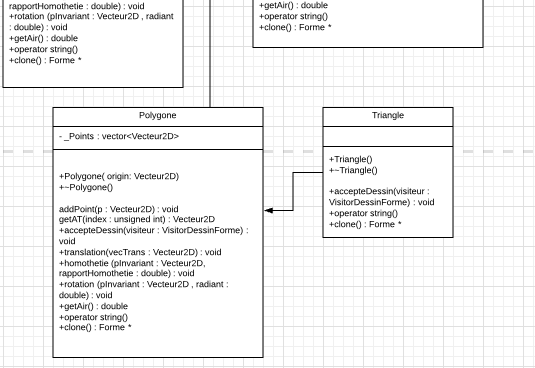
## 2. Forme géométrique

L'application gère deux sortes de formes : les formes géométriques simples et les formes géométriques composées.

* Une forme géométrique simple peut être un segment, un cercle, un triangle ou un polygone quelconque.
* Une forme géométrique composée est appelée groupe. Elle est composée d'une ou plusieurs formes géométriques (simples ou composées) disjointes.

Chaque Forme hérite de la class Forme qui définit toute les méthodes qu’une forme dois avoir. La class Triangle hérite d’un polygone car c’est un polygone particulier avec exactement 3 cotés ainsi on évite la redondance de code.





## 2.1 transformations géométriques

L'application offre la possibilité d'appliquer trois sortes de transformations géométriques aux formes : translation, homothétie et rotation. Dès qu’une transformation est appelé sur une forme, On envoi l’ordre à ces vecteurs qui effectue les transformations nécessaires et on transforme les autres champs de la forme si c’est necessaire . par exemple pour une homothétie sur un cercle il faut aussi transformer son rayon

## 2.3 String

Lorsque l’on converti une Forme en string on obtient l’exemple suivant :



Cercle : le type de la forme

Couleur:(0,100,255) : la Couleur de la forme sous le format RGB

Centre(100,100) : le centre du cercle représenter par un vecteur

Rayon :800 : le rayon du cercle représenter par un double

Chaque Forme a sa propre représentation selon les différent Champs quel a mais peu importe la forme tous ces champs son séparer par un tiret. Lorsque qu’une forme est représentée seulement par une liste de Vecteur on a simplement les vecteur les uns après les autre . exemple : (100,200)-(200,200)-(200,100)

Pour un groupe cela se présente comme suit :



D’abord le type puis la couleur et On ouvre les crochets et les différente forme du groupe sous séparer un « | »

C’est le format qui est utilisé lorsque l’on convertie la forme en string mais aussi lorsque que l’on sauvegarde la forme sur un fichier disque et l’expression régulière utiliser par la chaine de responsabilité qui convertit un string en forme lorsqu’on charge un fichier disque.

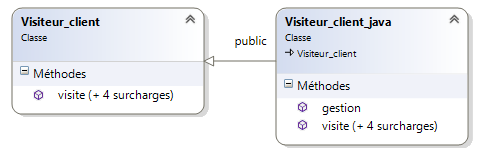
# 3. Dessiner une forme

## On utilise le DP Visitor pour dessiner les différente Forme. L’application implémente pour l’instant 2 class différente pour Dessiner en java et en sfml mais peu par la suite en implémenter des autres facilement.

## 3.1 Dessiner Java

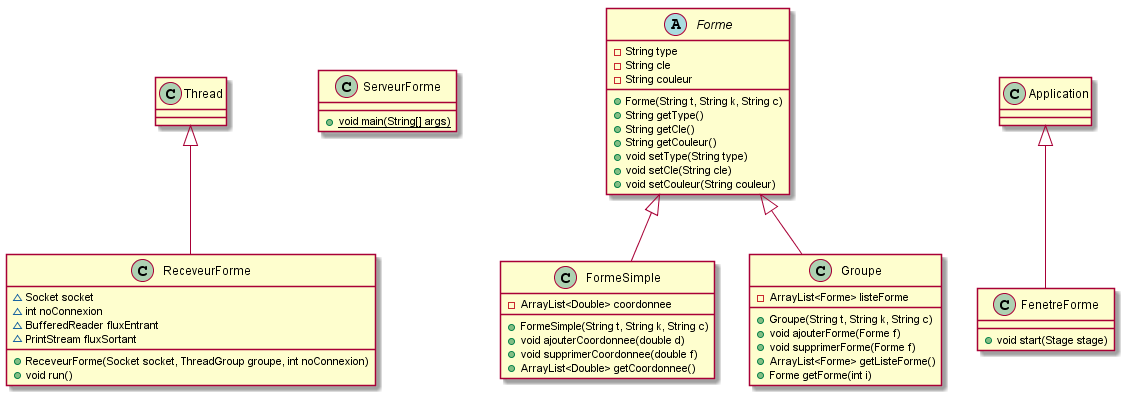
La méthode C++, dessine\_forme\_client\_java, est un client TCP/IP vers un serveur JAVA de dessin.

Grâce à la librairie Winsock, la méthode utilise une connexion TCP/IP vers le serveur ouvert au préalable, puis transmet successivement au serveur les instructions d'ouverture de fenêtre graphique, puis de tracés. Le serveur qui est multi-client, se charge d'exécuter les requêtes du client. Pour l'organisation de l'application distribuée. L’utilisation du Design Pattern Visitor permet de rendre cette méthode indépendante de la librairie graphique utilisée.

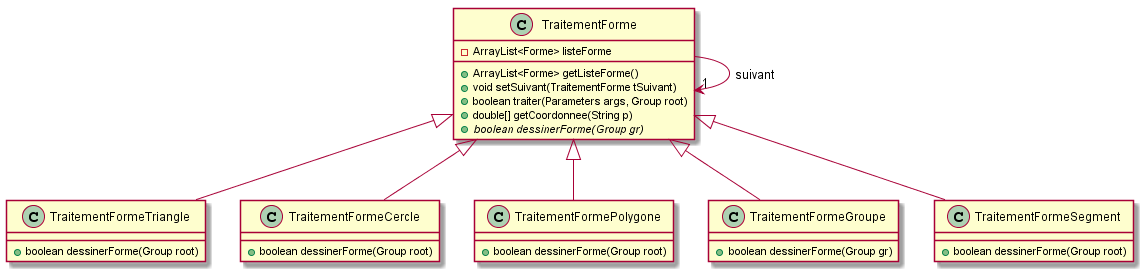


Côté serveur :

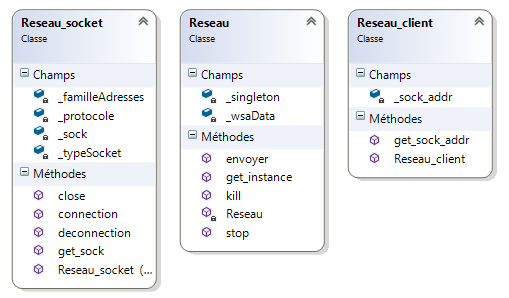
La classe serveurForme est la classe qui créer un serveur java. Elle lance une classe ReceveurForme qui va s’occuper de lancer le Chain of Responsability Traitement.



La classe mère TraitementForme effectue un traitement des requêtes reçue par un client grâce à sa méthode traiter(). Cette méthode transforme une chaine de caractère en un objet Forme Simple ou un Groupe et l’ajoute à sa liste de forme. Puis il suffit de créer la chaine de traitement afin que la liste soit traitée par la méthode dessiner Forme.



L'initialisation de la Winsock est effectuée par la classe Reseau qui est une classe Singleton de façon à garantir l’unicité de l'initialisation.

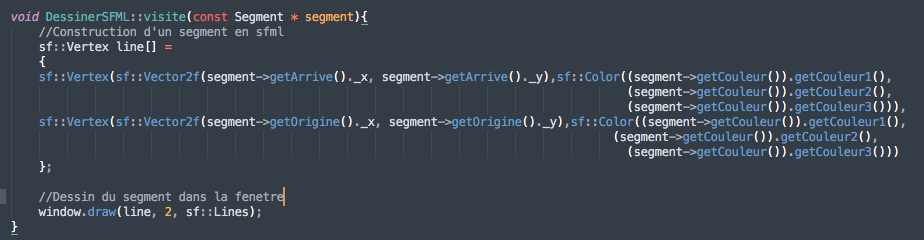


## 3.2 Dessiner SFML :

La classe DessinerSFML hérite de VisitorDessinerForme. Son constructeur prend en paramètre 2 int pour construire la fenêtre SFML. Et a donc un champs fenêtre

Chaque méthodes visite Construit la forme qu’il a en paramètre et la dessine dans la fenêtre.

Ex pour un segment :



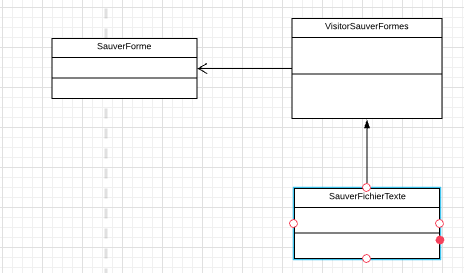
La méthode run affiche la fenêtre à l’écran.

# 4. Sauvegarde/Chargement de forme

Une forme peut être sauvegardée ou chargée à partir d'un fichier disque. Pour rendre les méthodes permettant cette opération indépendante, des design pattern ont encore une fois été utilisés. Ainsi il est aisément possible de rajouter ou changer les moyens de sauvegarder/charger .

## 4.1 Design Pattern Visitor

Le Design Pattern Visitor est utilisé pour la sauvegarde d'une forme. Pour l’instant l’application permet simplement de sauvegarder sur un fichier disque



## 4.2 Design Pattern Chain Of Responsibility

Le Design Pattern Chain Of Responsibility est utilisé pour le chargement d’une forme. La classe ChargeurListeFormes charge un fichier disque et envoi chaque ligne à la chaine de responsabilité qui traite la ligne et renvoi une forme si cela est possible .

